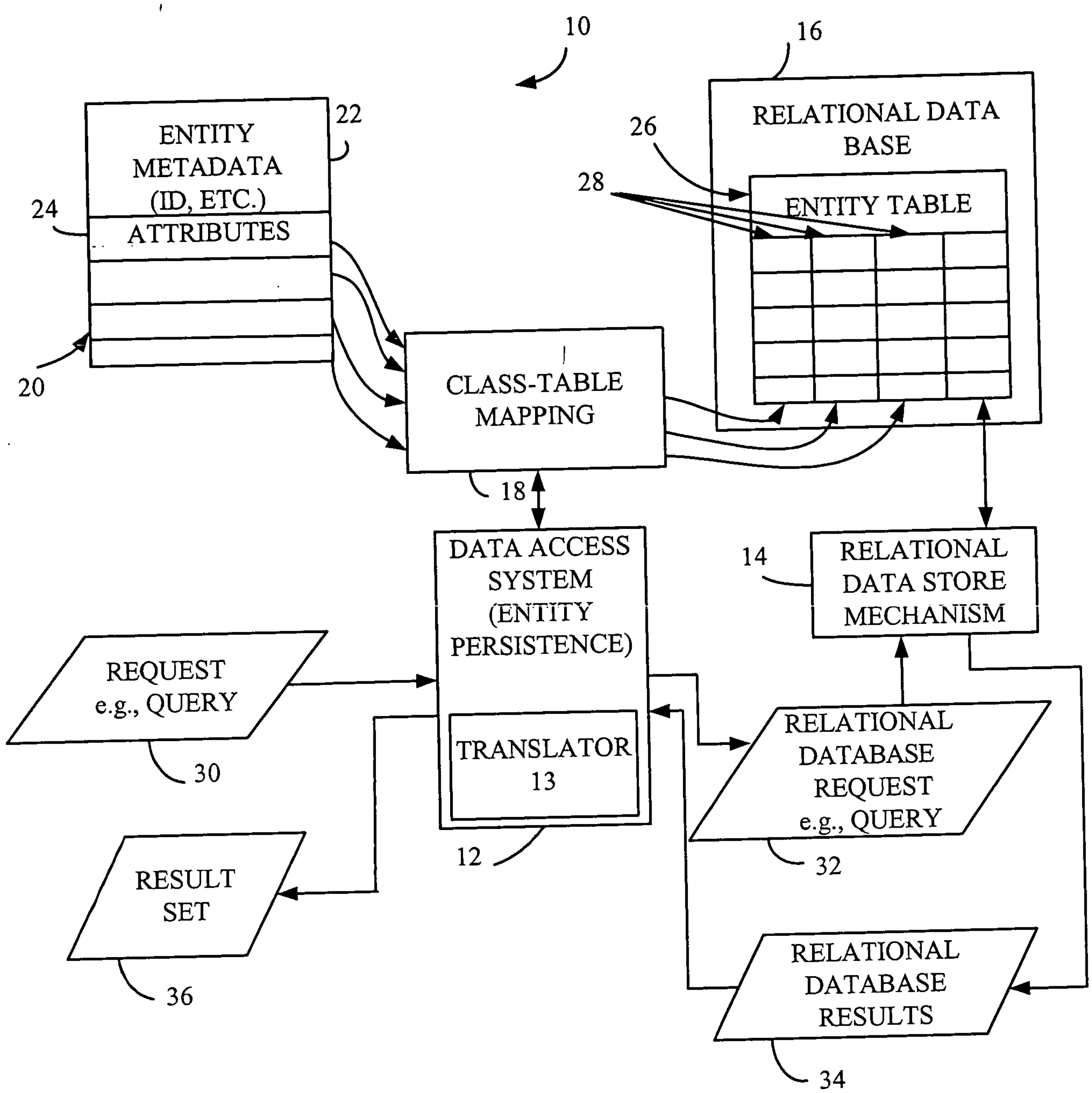
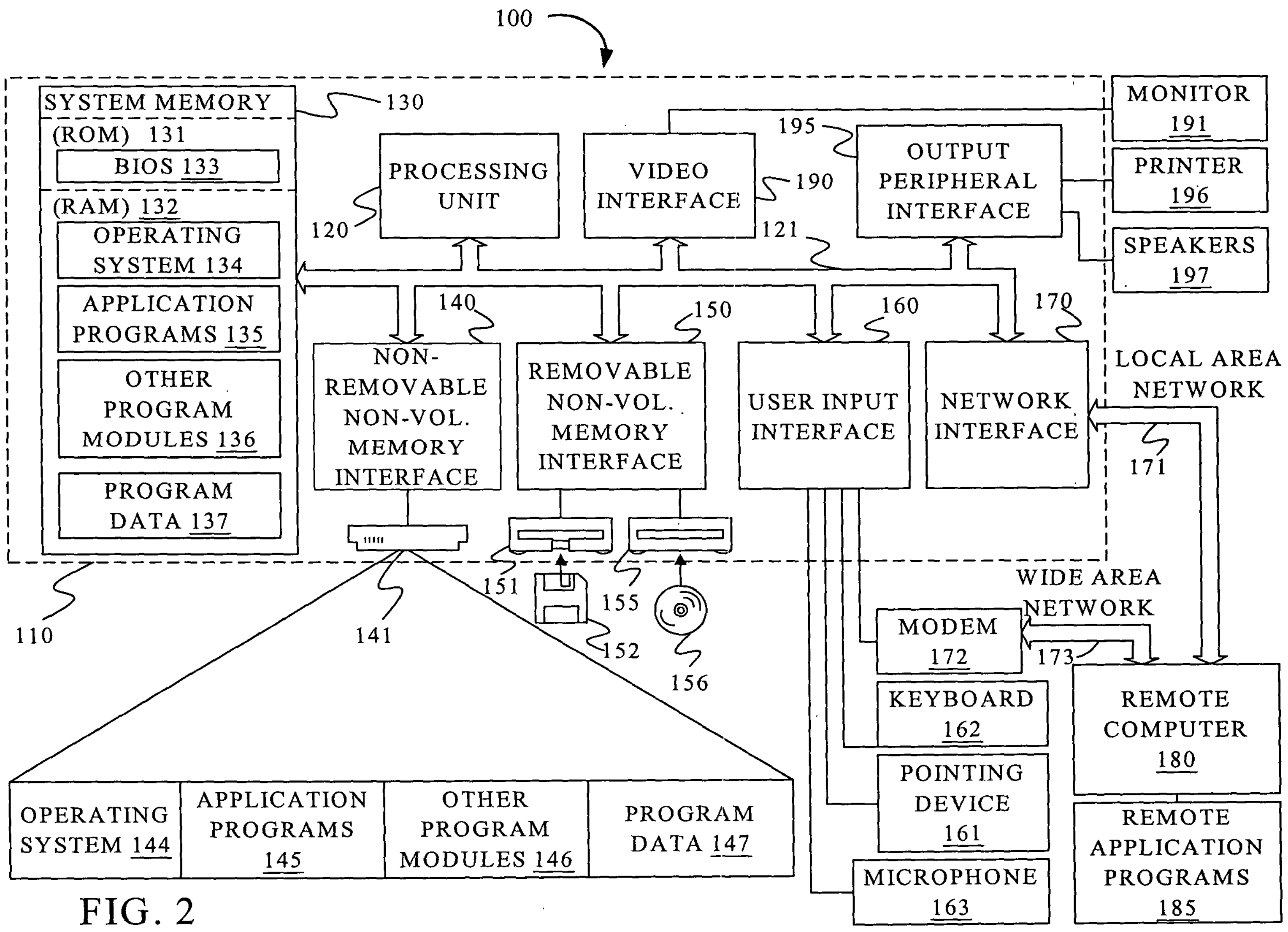
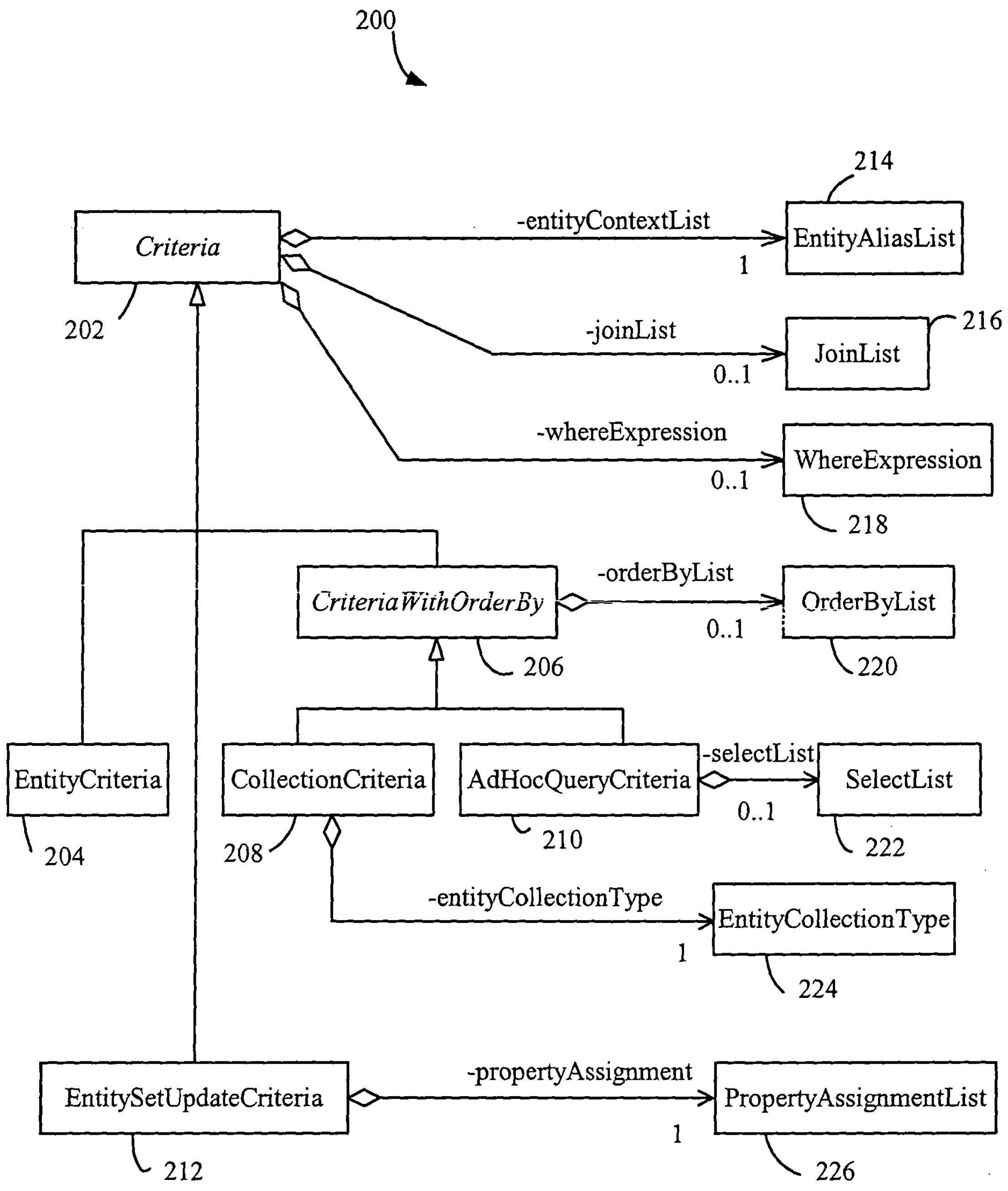

10
16
22
ENTITY METADATA (ID, ETC.)
24
ATTRIBUTES
20
26
RELATIONAL DATA BASE
28
ENTITY TABLE
CLASS-TABLE MAPPING
18
DATA ACCESS SYSTEM (ENTITY PERSISTENCE)
TRANSLATOR 13
12
14
RELATIONAL DATA STORE MECHANISM
REQUEST e.g., QUERY
30
RESULT SET
36
RELATIONAL DATABASE REQUEST e.g., QUERY
32
RELATIONAL DATABASE RESULTS
34

FIG. 1


100
SYSTEM MEMORY
130
(ROM) 131
BIOS 133
(RAM) 132
OPERATING SYSTEM 134
APPLICATION PROGRAMS 135
OTHER PROGRAM MODULES 136
PROGRAM DATA 137
PROCESSING UNIT
120
VIDEO INTERFACE
190
OUTPUT PERIPHERAL INTERFACE
195
121
MONITOR 191
PRINTER 196
SPEAKERS 197
NON-REMOVABLE NON-VOL. MEMORY INTERFACE
140
REMOVABLE NON-VOL. MEMORY INTERFACE
150
USER INPUT INTERFACE
160
NETWORK INTERFACE
170
LOCAL AREA NETWORK
171
110
141
151
152
155
156
WIDE AREA NETWORK
MODEM 172
173
KEYBOARD 162
POINTING DEVICE 161
MICROPHONE 163
REMOTE COMPUTER 180
REMOTE APPLICATION PROGRAMS 185
OPERATING SYSTEM 144
APPLICATION PROGRAMS 145
OTHER PROGRAM MODULES 146
PROGRAM DATA 147
FIG. 2



200
Criteria
202
-entityContextList
1
EntityAliasList
214
-joinList
0..1
JoinList
216
-whereExpression
0..1
WhereExpression
218
CriteriaWithOrderBy
206
-orderByList
0..1
OrderByList
220
EntityCriteria
204
CollectionCriteria
208
AdHocQueryCriteria
210
-selectList
0..1
SelectList
222
-entityCollectionType
1
EntityCollectionType
224
EntitySetUpdateCriteria
212
-propertyAssignment
1
PropertyAssignmentList
226

FIG. 3

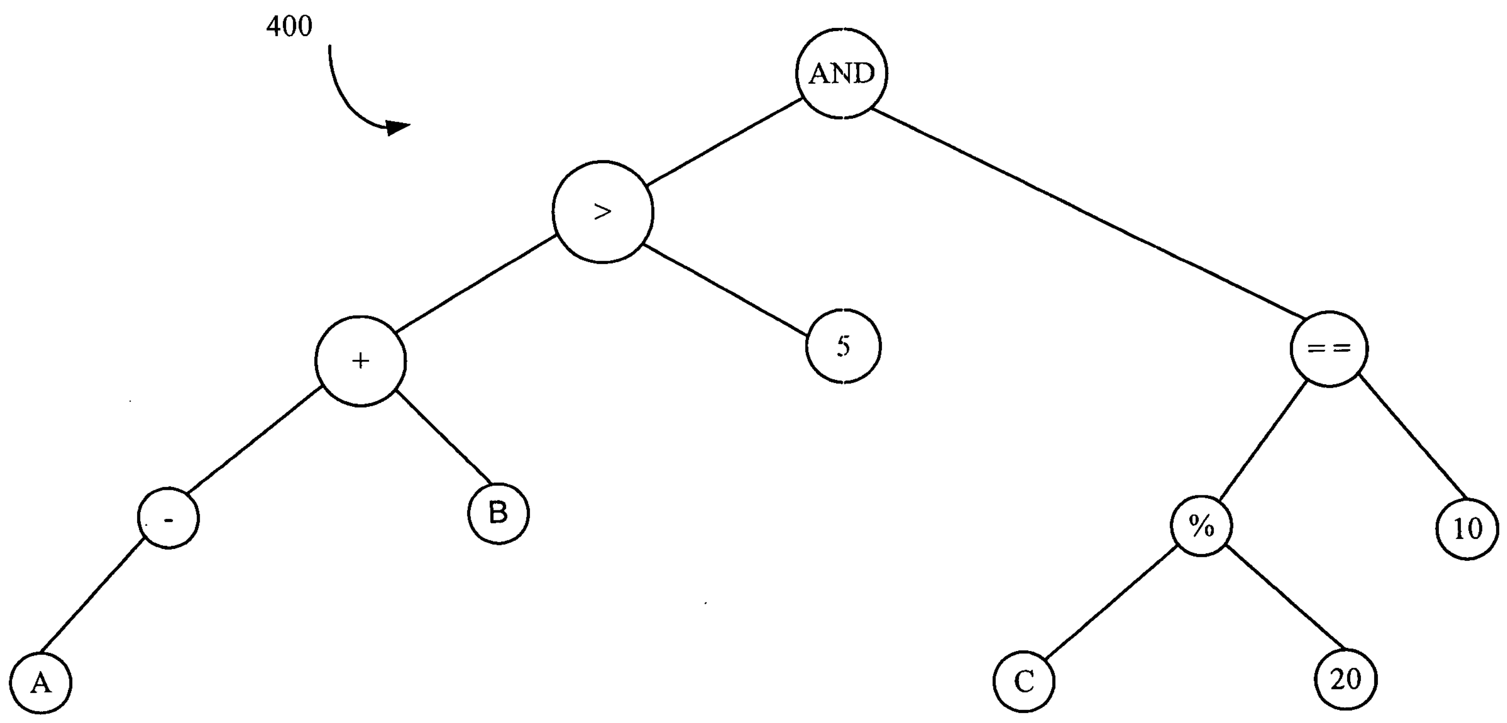


FIG. 4A

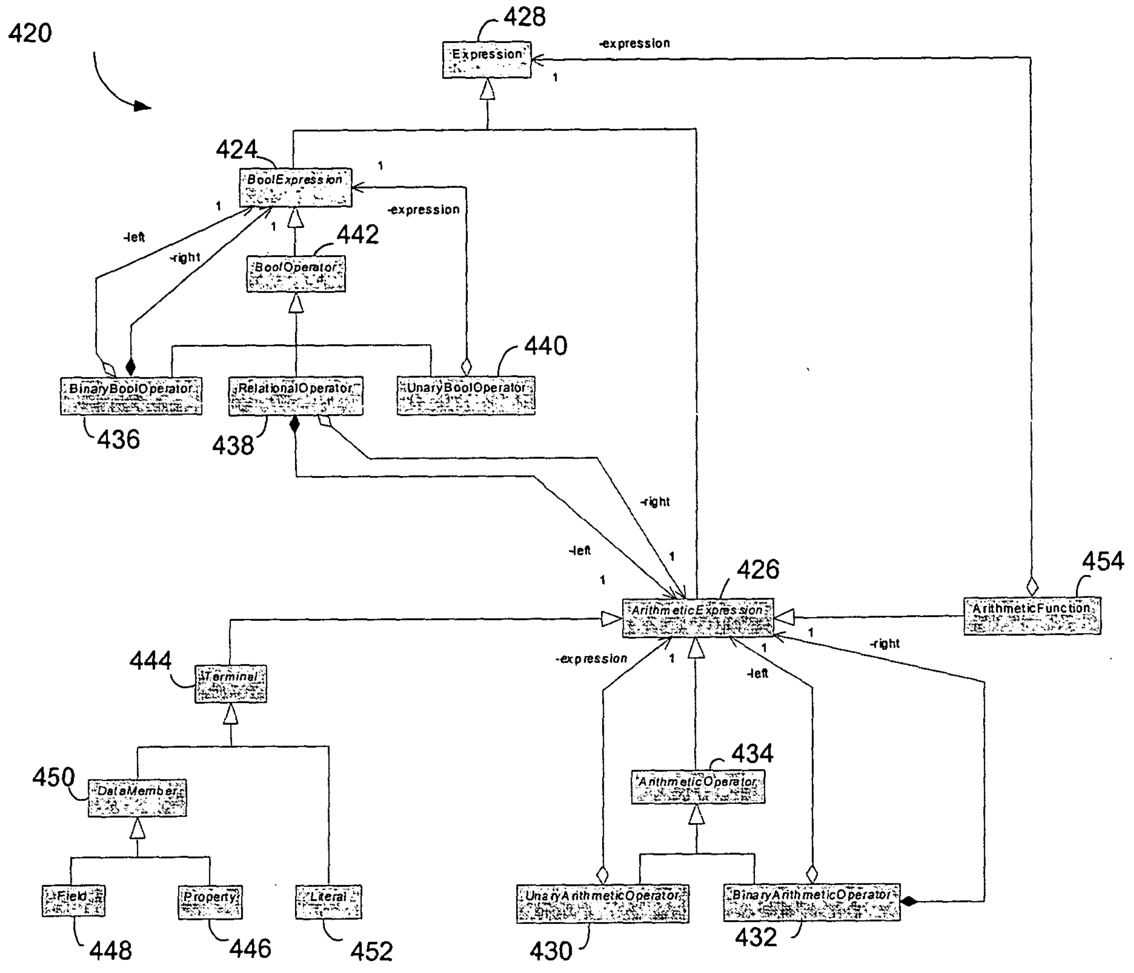

420
428
Expression
-expression
1
424
BoolExpression
1
1
-expression
-left
-right
1
442
BoolOperator
440
BinaryBoolOperator
RelationalOperator
UnaryBoolOperator
436
438
-right
-left
1
1
426
454
ArithmeticExpression
ArithmeticFunction
1
-right
-expression
1
-left
1
444
Terminal
434
ArithmeticOperator
450
DataMember
Field
Property
Literal
UnaryArithmeticOperator
BinaryArithmeticOperator
448
446
452
430
432

FIG. 4B

FIG. 4C

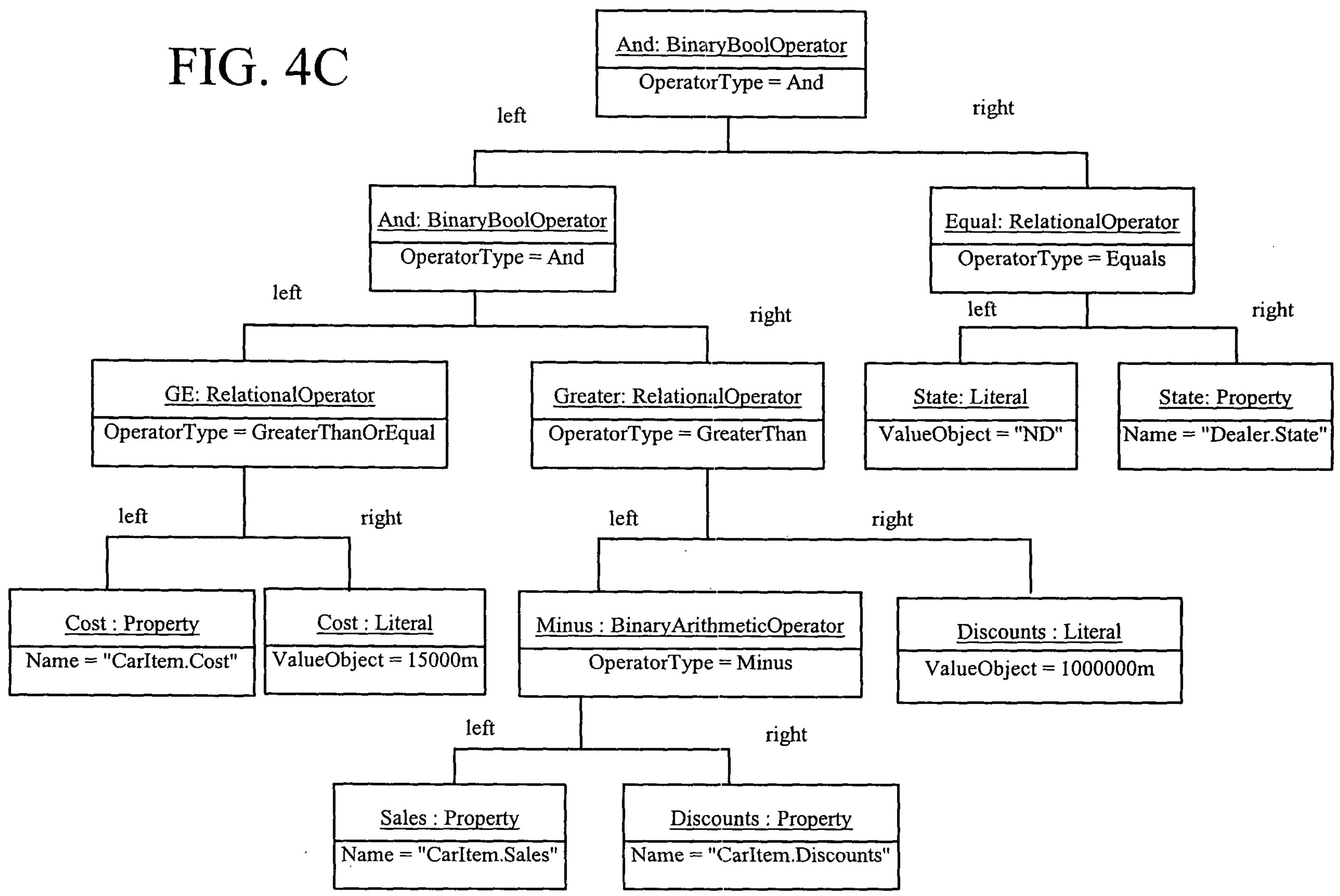
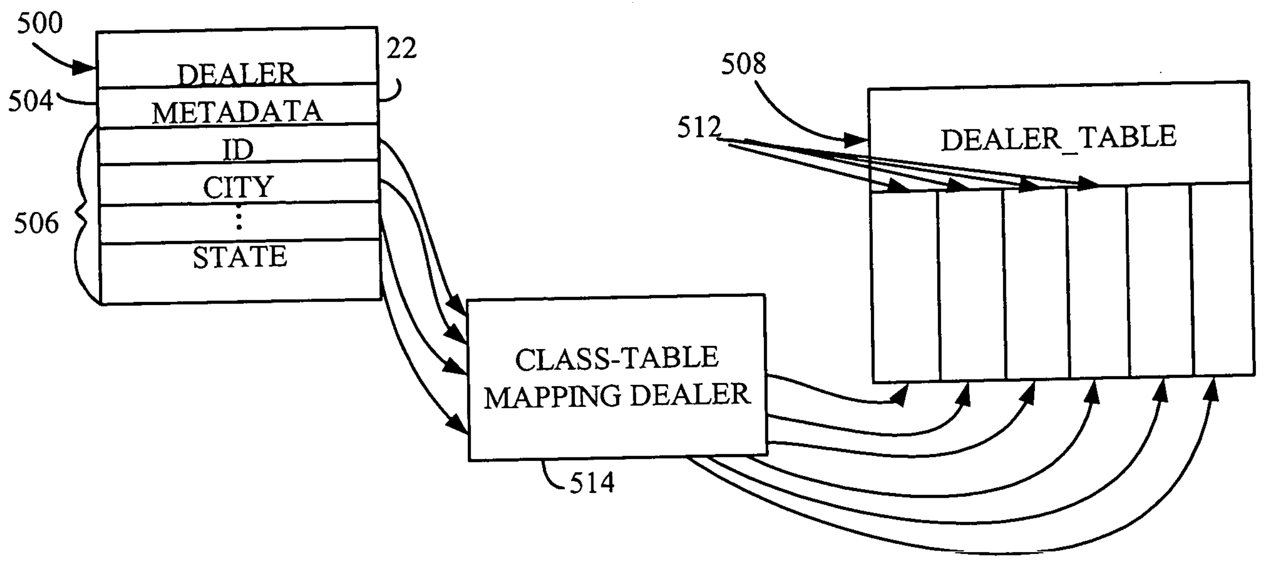
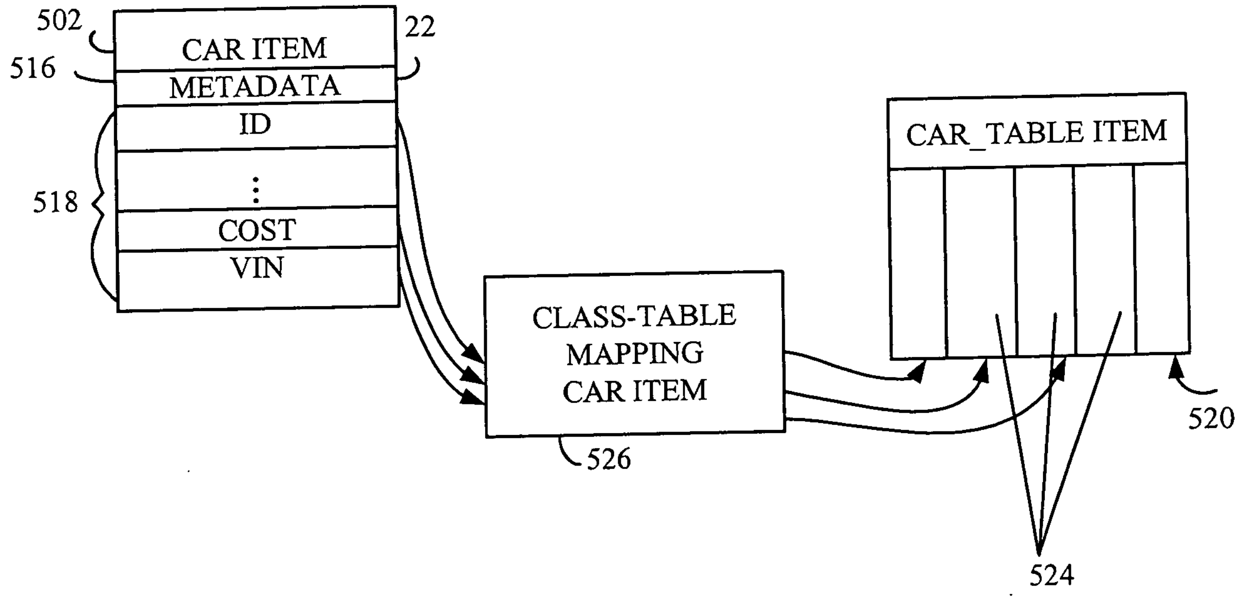

And: BinaryBoolOperator
OperatorType = And
left
right
And: BinaryBoolOperator
OperatorType = And
Equal: RelationalOperator
OperatorType = Equals
left
right
left
right
GE: RelationalOperator
OperatorType = GreaterThanOrEqual
Greater: RelationalOperator
OperatorType = GreaterThan
State: Literal
ValueObject = "ND"
State: Property
Name = "Dealer.State"
left
right
left
right
Cost : Property
Name = "CarItem.Cost"
Cost : Literal
ValueObject = 15000m
Minus : BinaryArithmeticOperator
OperatorType = Minus
Discounts : Literal
ValueObject = 1000000m
left
right
Sales : Property
Name = "CarItem.Sales"
Discounts : Property
Name = "CarItem.Discounts"

FIG. 5


500
DEALER
22
504
METADATA
ID
CITY
STATE
506
508
512
DEALER_TABLE
CLASS-TABLE MAPPING DEALER
514

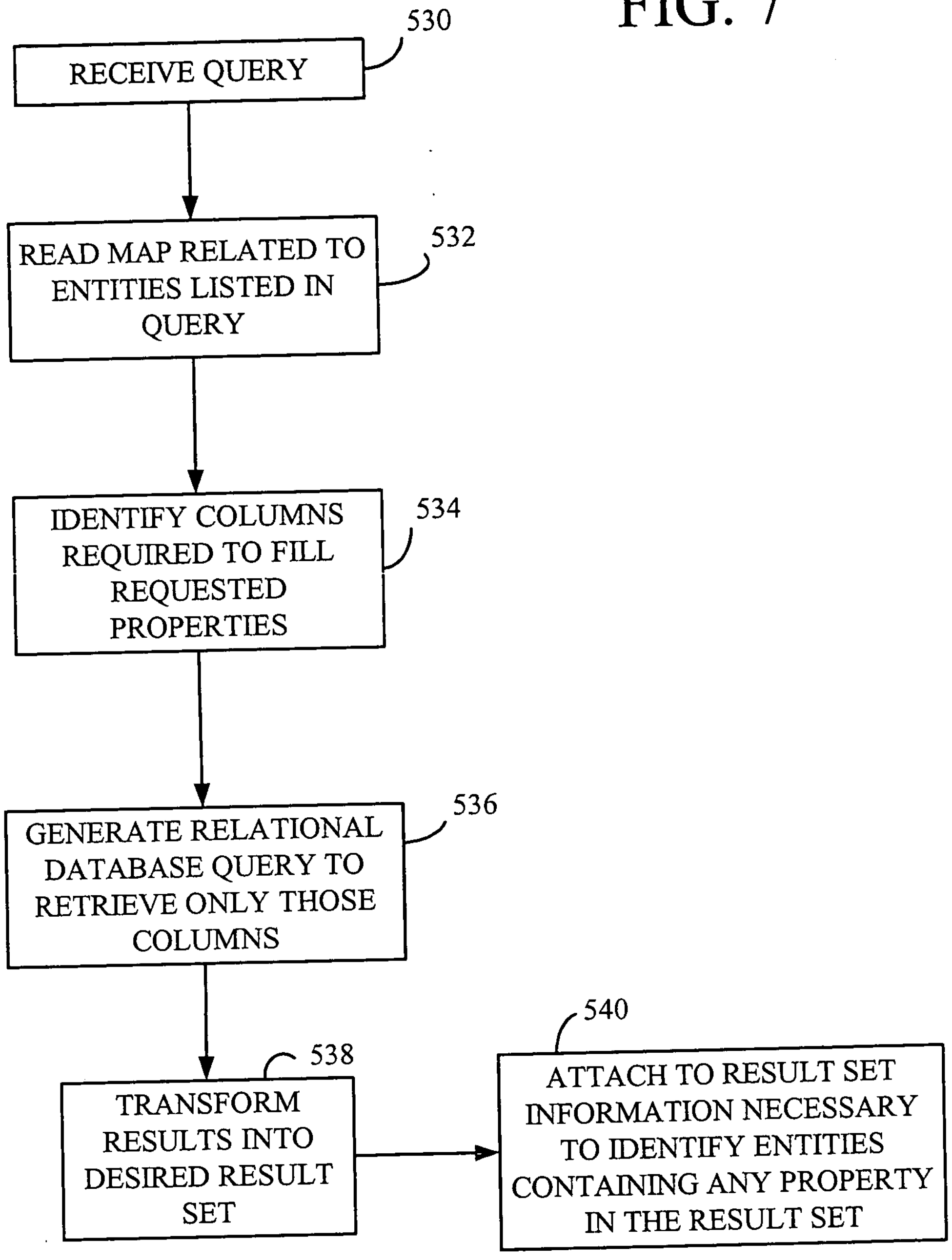

502
CAR ITEM
22
516
METADATA
ID
COST
VIN
518
CAR_TABLE ITEM
CLASS-TABLE MAPPING CAR ITEM
526
520
524

```
class CarItem {            // these properties are mapped to a database table
        public string ID
        public string Vin;
        public string Color;
        public decimal Cost;
        ... many others omitted ...
}
class Dealer {             // these properties are mapped to a different database table
        public string ID;
        public string City;
        public string State;
        ... many others omitted ...
}

AdHocQueryCriteria  adHocCriteria = Criteria.AdHocQueryCriteria(
        Criteria.EntityAliases(    // describes the objects involved in the query
                Criteria.EntityAlias(itemParentKey, typeof(CarItem))
                Criteria.EntityAlias(dealerParentKey, typeof(Dealer)) ),
        Criteria.JoinList(
                        /* entity to entity join */
                Criteria.InnerJoin("CarItem", "Dealer",
                (Property)"CarItem.DealerID" == (Property)"Dealer.ID")),
        Criteria.Select(            // the specific properties to retrieve
                (Property)"CarItem.ID",  // references the field in the above class
                (Property)"CarItem.Vin",
                (Property)"CarItem.Cost",
                (Property)"Dealer.ID",
                (Property)"Dealer.City",
                (Property)"Dealer.State"),
        Criteria.Where(
                (Property)"CarItem.Make" == "Geo" &&
                (Property)"CarItem.Model" == "Prism" &&
                (Property)"Dealer.State" == "ND")
        Criteria.OrderBy((Property)"Dealer.Cost"));
```

FIG. 6

FIG. 7


530
RECEIVE QUERY
532
READ MAP RELATED TO ENTITIES LISTED IN QUERY
534
IDENTIFY COLUMNS REQUIRED TO FILL REQUESTED PROPERTIES
536
GENERATE RELATIONAL DATABASE QUERY TO RETRIEVE ONLY THOSE COLUMNS
538
TRANSFORM RESULTS INTO DESIRED RESULT SET
540
ATTACH TO RESULT SET INFORMATION NECESSARY TO IDENTIFY ENTITIES CONTAINING ANY PROPERTY IN THE RESULT SET

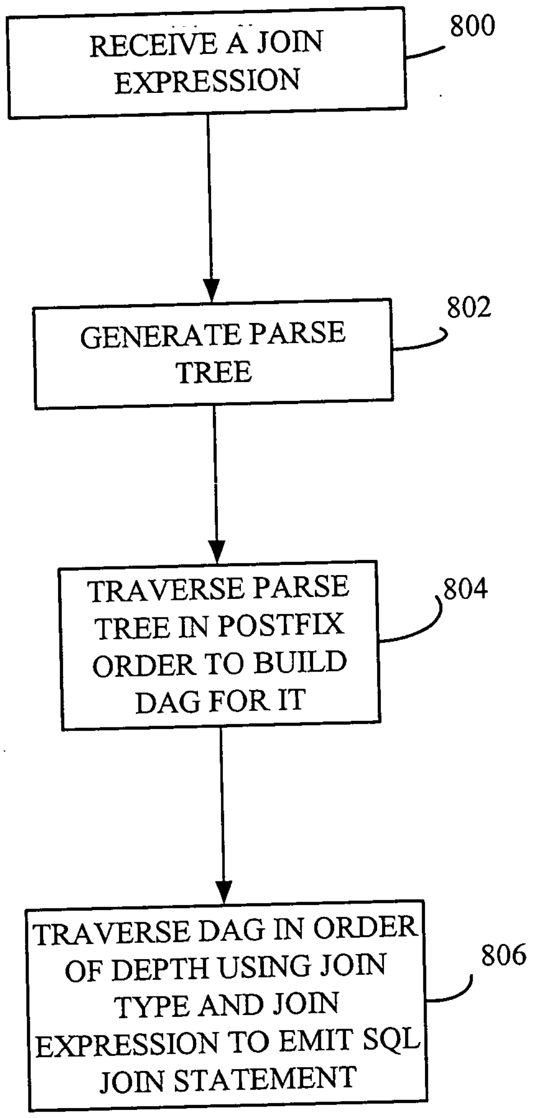

RECEIVE A JOIN EXPRESSION
800
GENERATE PARSE TREE
802
TRAVERSE PARSE TREE IN POSTFIX ORDER TO BUILD DAG FOR IT
804
TRAVERSE DAG IN ORDER OF DEPTH USING JOIN TYPE AND JOIN EXPRESSION TO EMIT SQL JOIN STATEMENT
806

FIG. 8

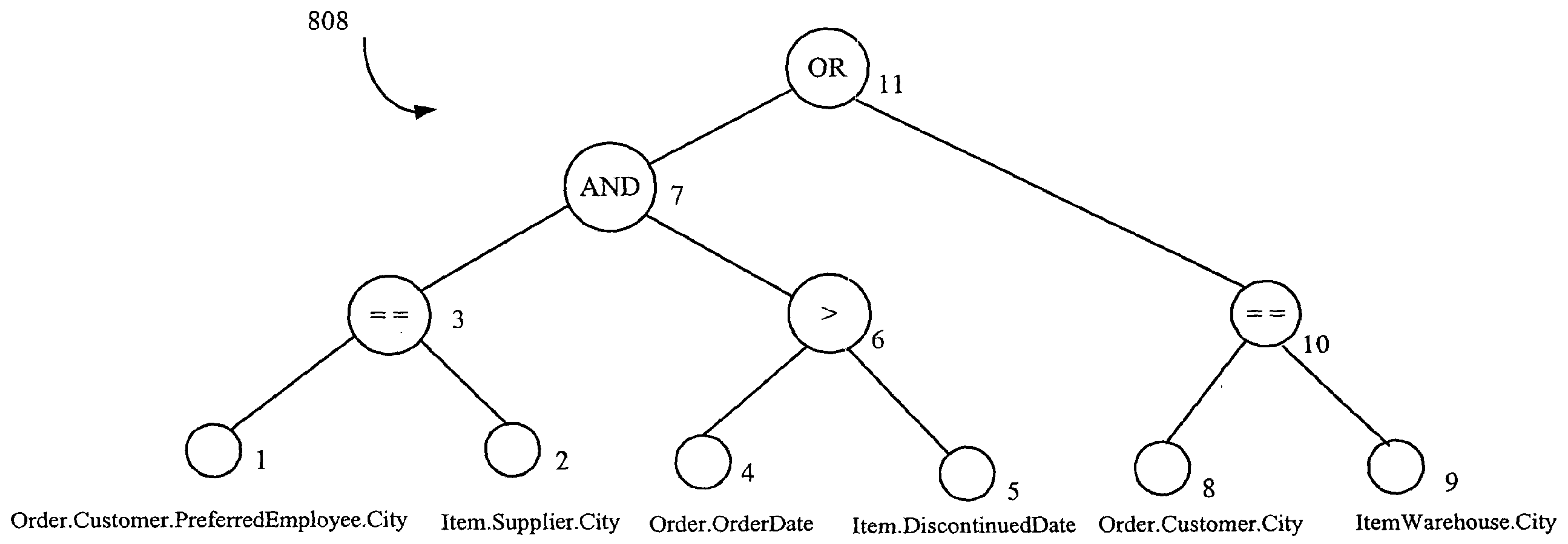
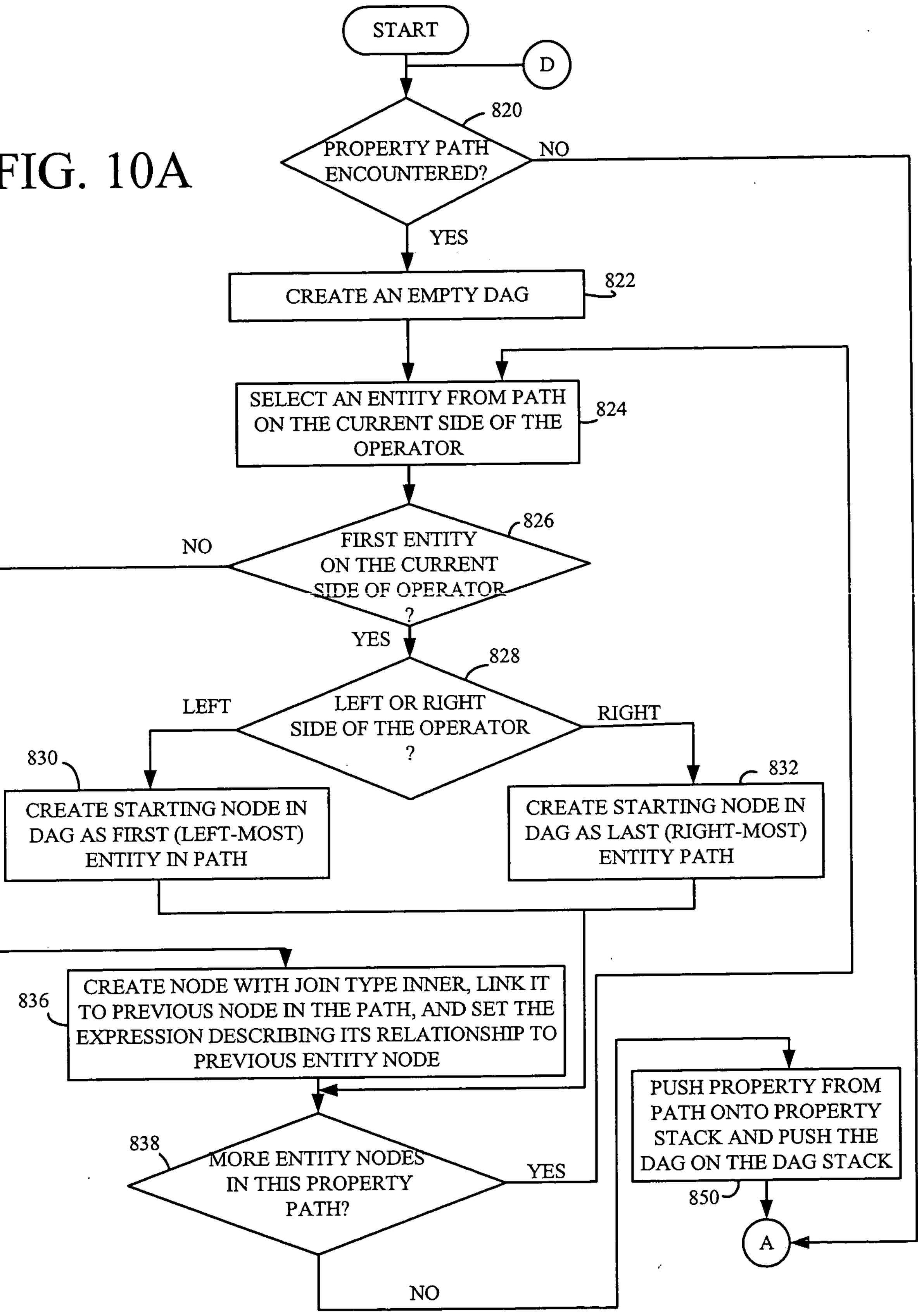

808
OR
11
AND
7
==
3
>
6
==
10
1
2
4
5
8
9
Order.Customer.PreferredEmployee.City
Item.Supplier.City
Order.OrderDate
Item.DiscontinuedDate
Order.Customer.City
ItemWarehouse.City

FIG. 9

FIG. 10A



START
D
820
PROPERTY PATH ENCOUNTERED?
NO
YES
CREATE AN EMPTY DAG
822
SELECT AN ENTITY FROM PATH ON THE CURRENT SIDE OF THE OPERATOR
824
826
FIRST ENTITY ON THE CURRENT SIDE OF OPERATOR ?
NO
YES
828
LEFT OR RIGHT SIDE OF THE OPERATOR ?
LEFT
RIGHT
830
CREATE STARTING NODE IN DAG AS FIRST (LEFT-MOST) ENTITY IN PATH
832
CREATE STARTING NODE IN DAG AS LAST (RIGHT-MOST) ENTITY PATH
836
CREATE NODE WITH JOIN TYPE INNER, LINK IT TO PREVIOUS NODE IN THE PATH, AND SET THE EXPRESSION DESCRIBING ITS RELATIONSHIP TO PREVIOUS ENTITY NODE
PUSH PROPERTY FROM PATH ONTO PROPERTY STACK AND PUSH THE DAG ON THE DAG STACK
838
MORE ENTITY NODES IN THIS PROPERTY PATH?
YES
850
A
NO

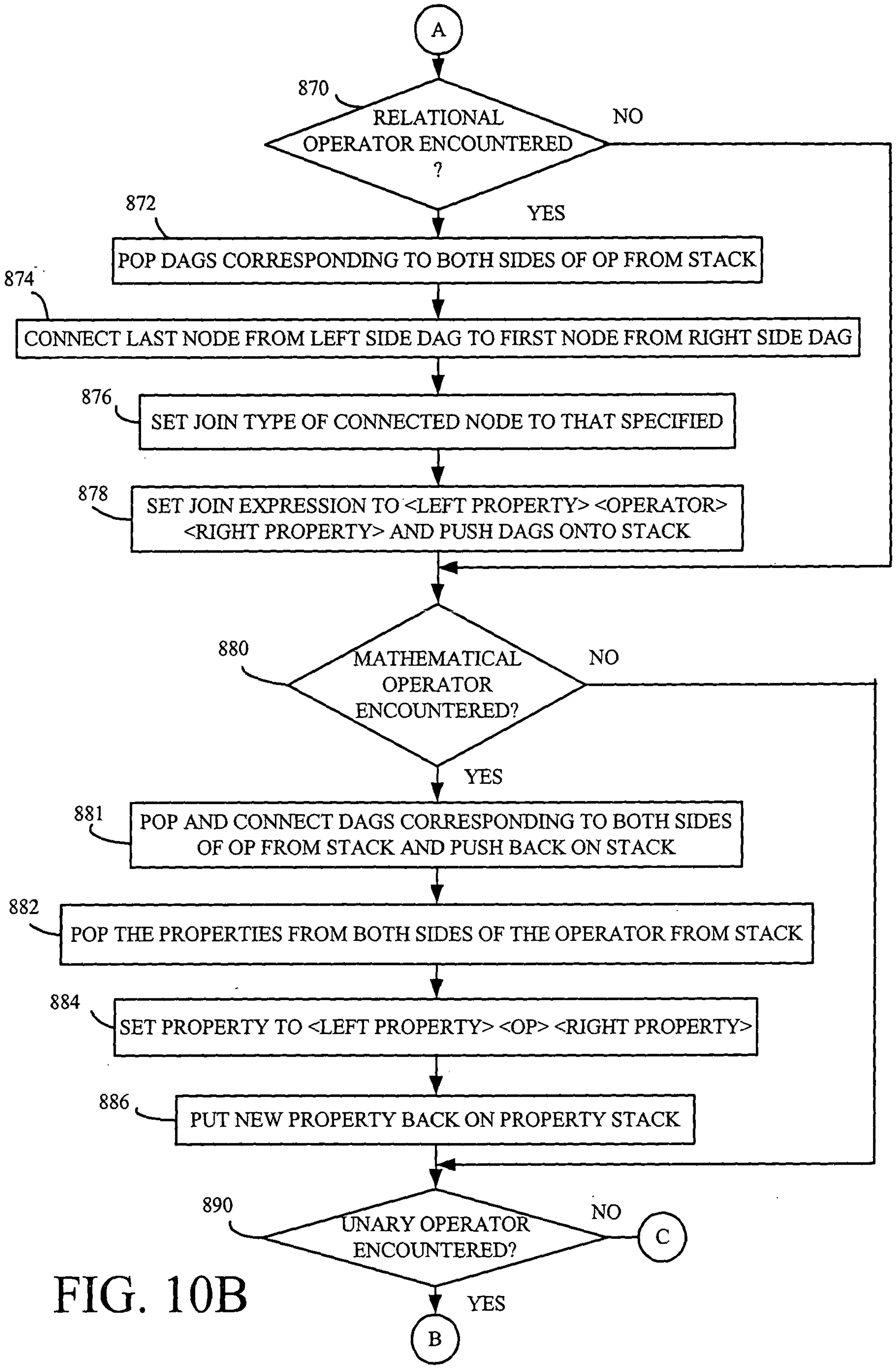
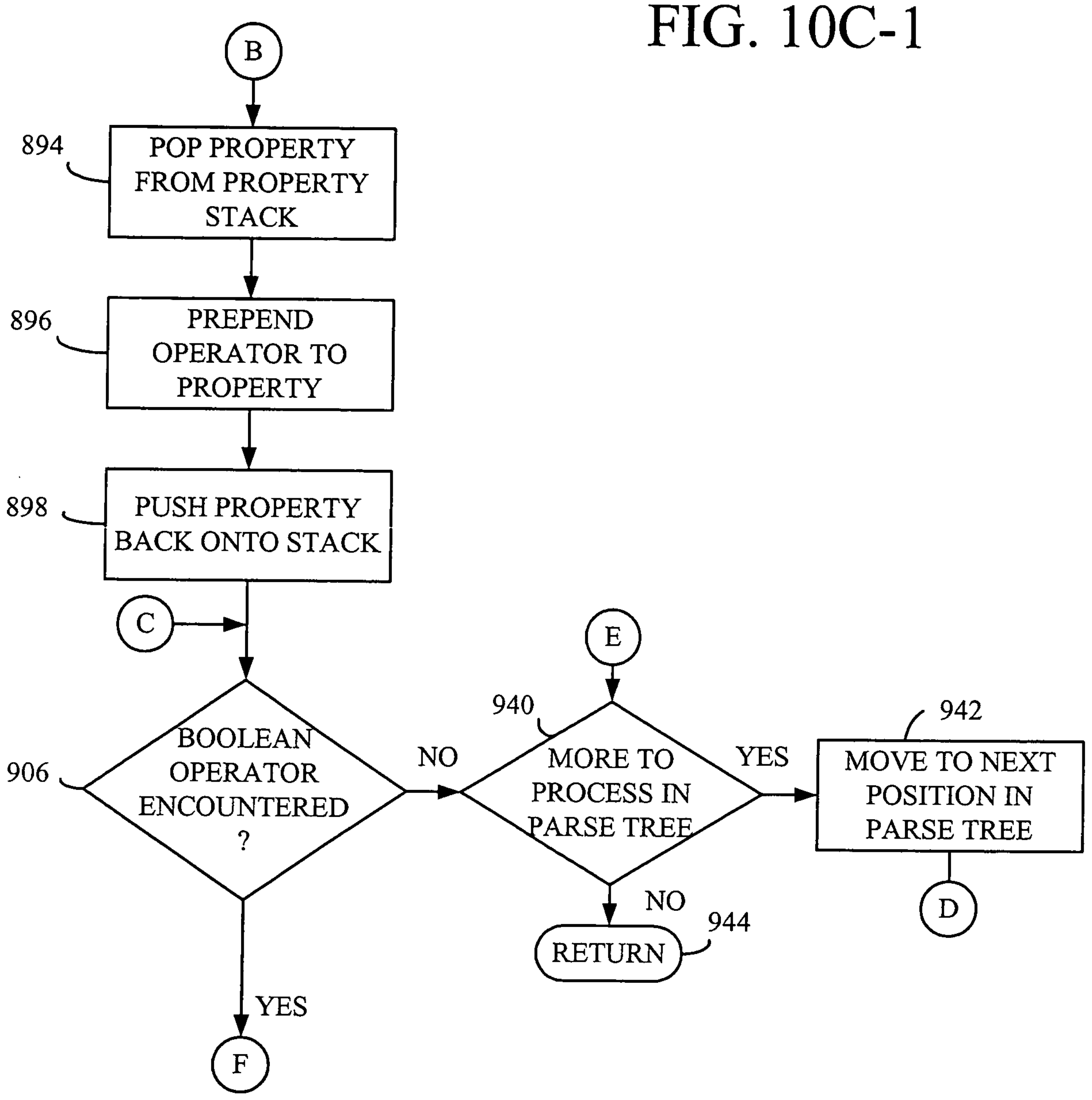

A
870
RELATIONAL OPERATOR ENCOUNTERED ?
NO
YES
872
POP DAGS CORRESPONDING TO BOTH SIDES OF OP FROM STACK
874
CONNECT LAST NODE FROM LEFT SIDE DAG TO FIRST NODE FROM RIGHT SIDE DAG
876
SET JOIN TYPE OF CONNECTED NODE TO THAT SPECIFIED
878
SET JOIN EXPRESSION TO <LEFT PROPERTY> <OPERATOR> <RIGHT PROPERTY> AND PUSH DAGS ONTO STACK
880
MATHEMATICAL OPERATOR ENCOUNTERED?
NO
YES
881
POP AND CONNECT DAGS CORRESPONDING TO BOTH SIDES OF OP FROM STACK AND PUSH BACK ON STACK
882
POP THE PROPERTIES FROM BOTH SIDES OF THE OPERATOR FROM STACK
884
SET PROPERTY TO <LEFT PROPERTY> <OP> <RIGHT PROPERTY>
886
PUT NEW PROPERTY BACK ON PROPERTY STACK
890
UNARY OPERATOR ENCOUNTERED?
NO
C
YES
B

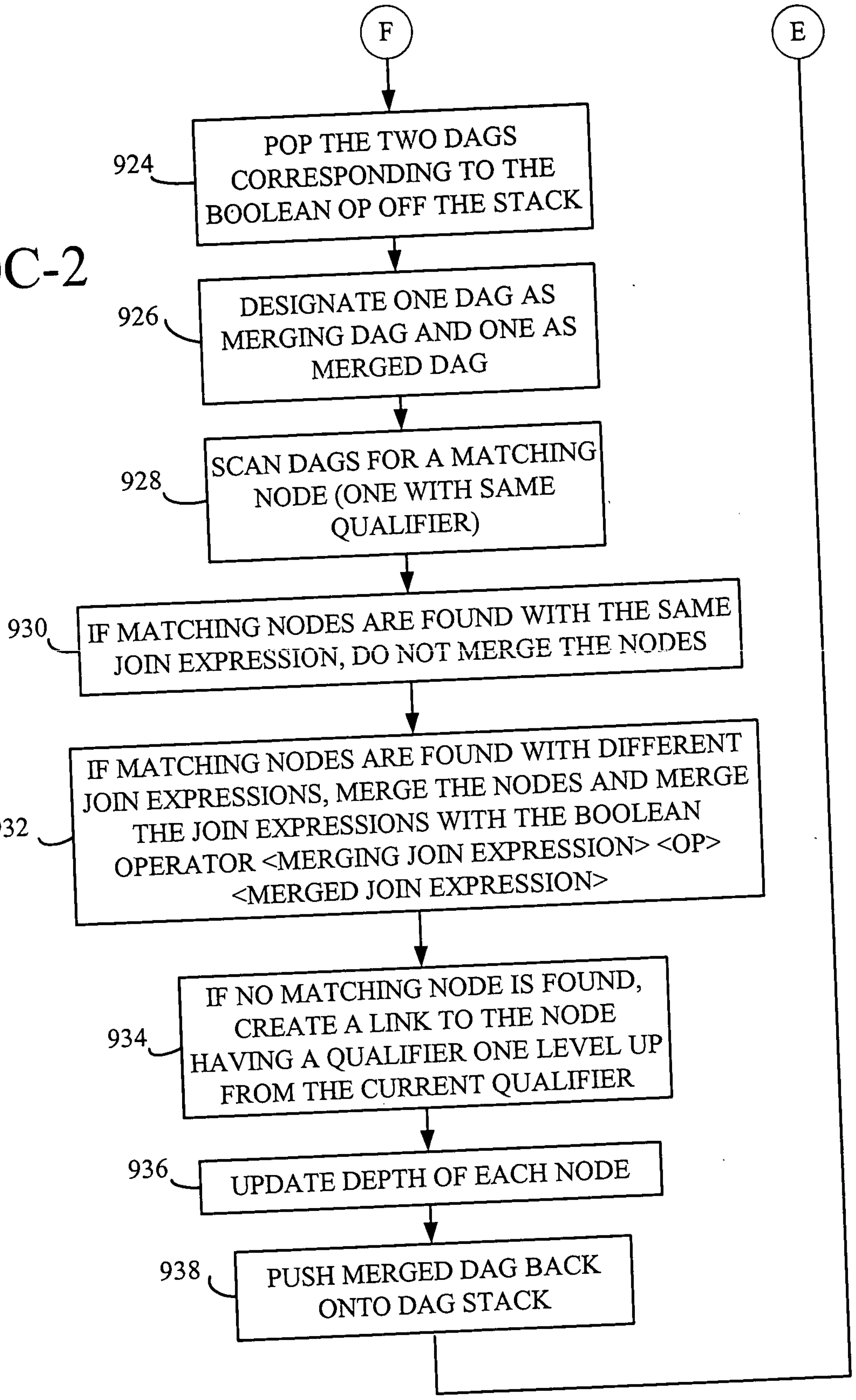
FIG. 10B

FIG. 10C-1



B
894
POP PROPERTY FROM PROPERTY STACK
896
PREPEND OPERATOR TO PROPERTY
898
PUSH PROPERTY BACK ONTO STACK
C
906
BOOLEAN OPERATOR ENCOUNTERED ?
NO
YES
F
E
940
MORE TO PROCESS IN PARSE TREE
YES
NO
944
RETURN
942
MOVE TO NEXT POSITION IN PARSE TREE
D

FIG. 10C-2



F
E
924
POP THE TWO DAGS CORRESPONDING TO THE BOOLEAN OP OFF THE STACK
926
DESIGNATE ONE DAG AS MERGING DAG AND ONE AS MERGED DAG
928
SCAN DAGS FOR A MATCHING NODE (ONE WITH SAME QUALIFIER)
930
IF MATCHING NODES ARE FOUND WITH THE SAME JOIN EXPRESSION, DO NOT MERGE THE NODES
932
IF MATCHING NODES ARE FOUND WITH DIFFERENT JOIN EXPRESSIONS, MERGE THE NODES AND MERGE THE JOIN EXPRESSIONS WITH THE BOOLEAN OPERATOR <MERGING JOIN EXPRESSION> <OP> <MERGED JOIN EXPRESSION>
934
IF NO MATCHING NODE IS FOUND, CREATE A LINK TO THE NODE HAVING A QUALIFIER ONE LEVEL UP FROM THE CURRENT QUALIFIER
936
UPDATE DEPTH OF EACH NODE
938
PUSH MERGED DAG BACK ONTO DAG STACK

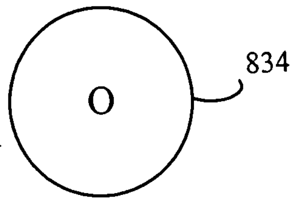


FIG. 11A

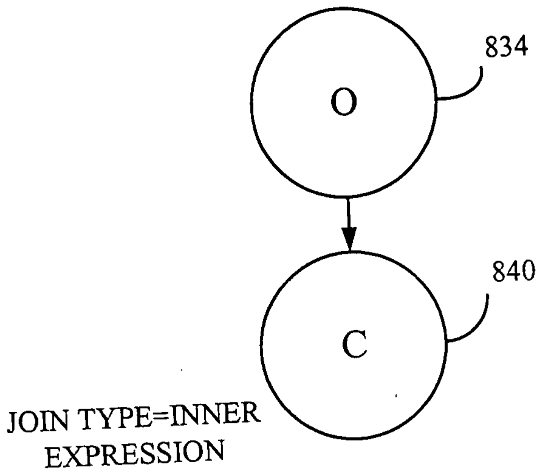


FIG. 11B

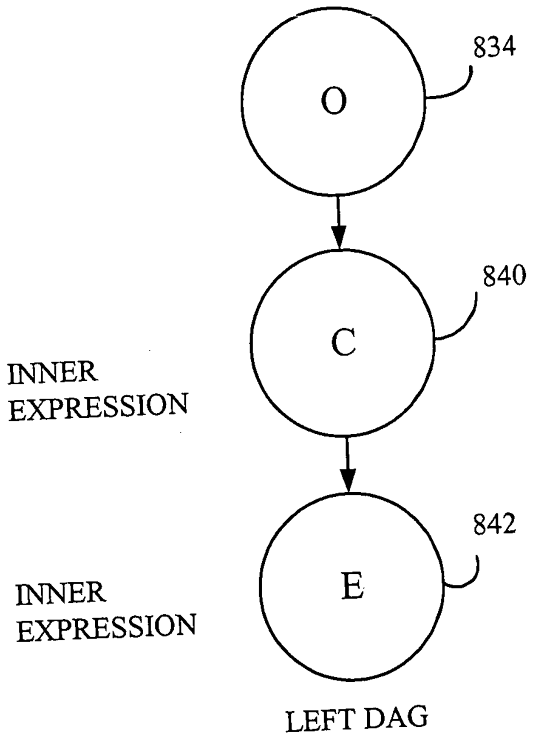


FIG. 11C

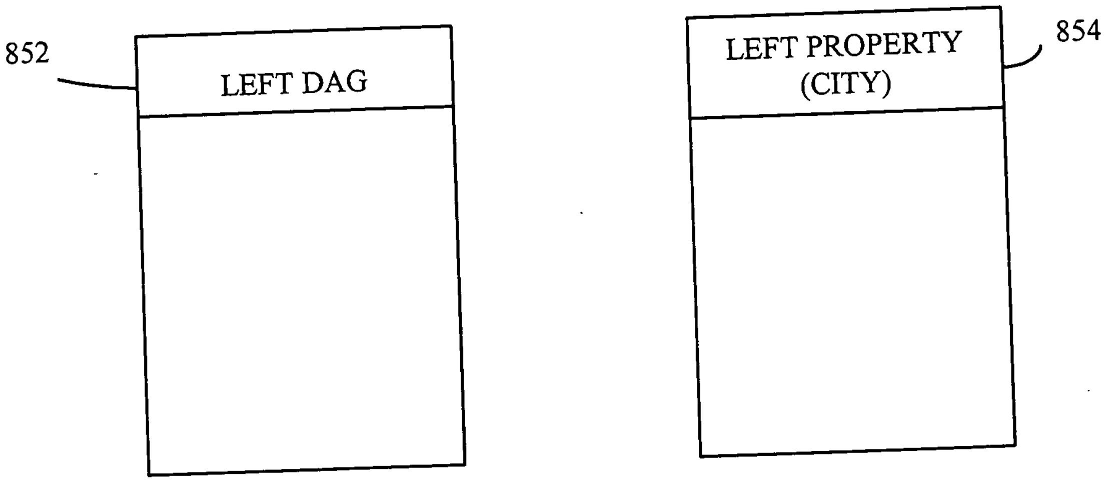


FIG. 11D

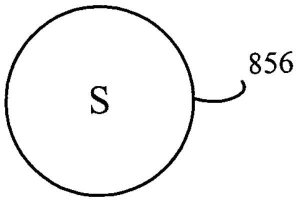


FIG. 11E

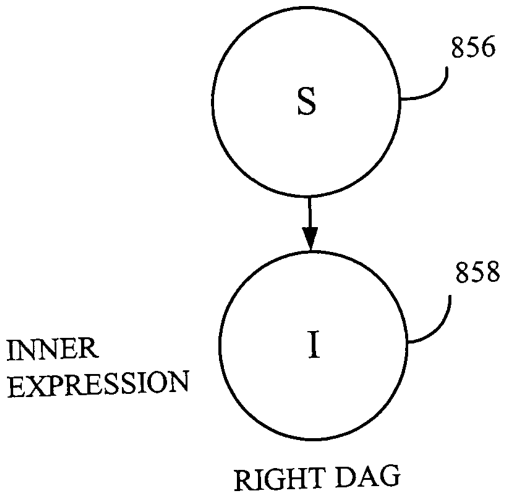


FIG. 11F

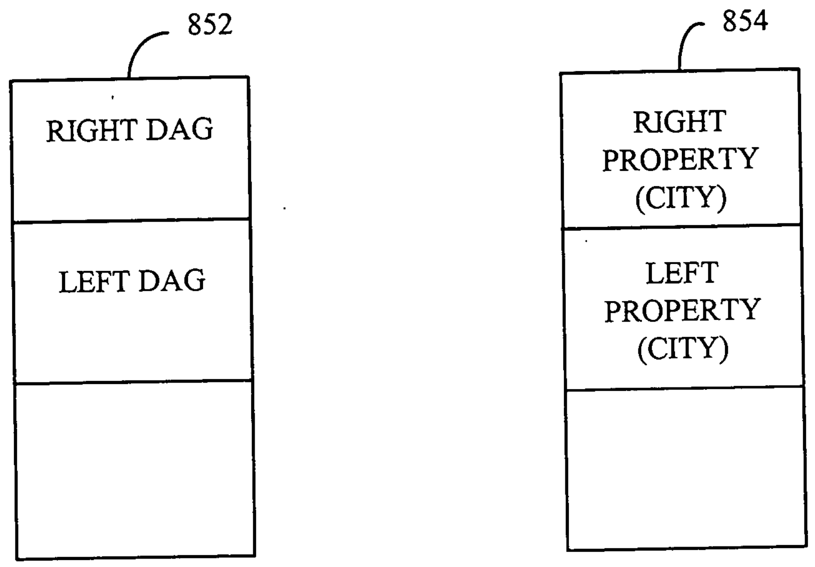


FIG. 11G

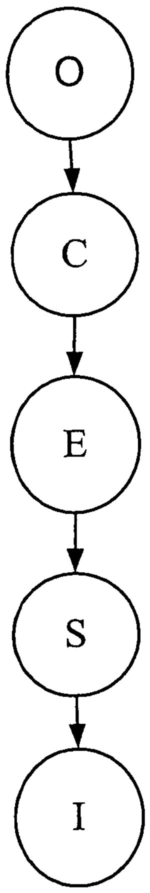


FIG. 12

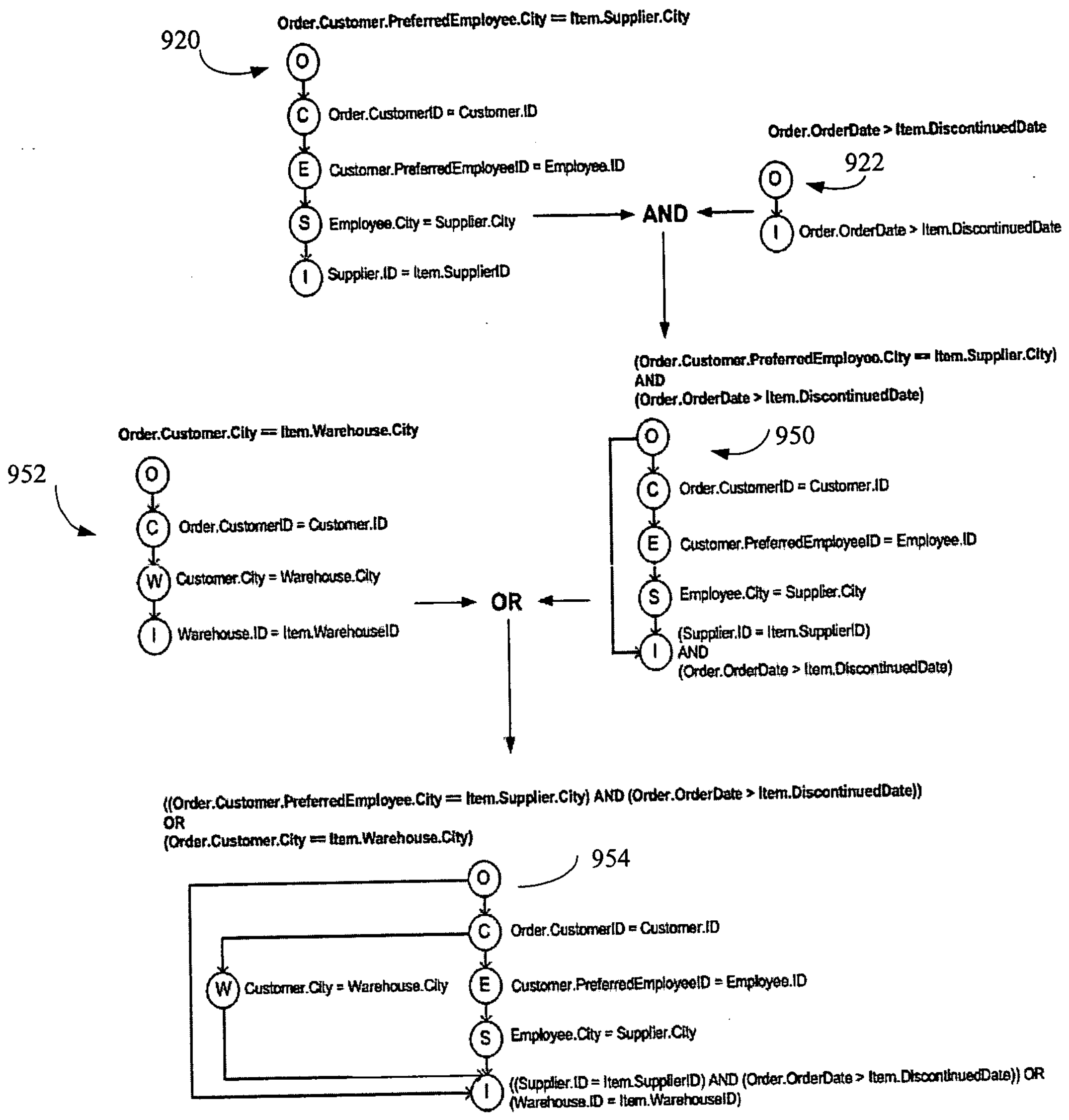

Order.Customer.PreferredEmployee.City == Item.Supplier.City
920
O
C
Order.CustomerID = Customer.ID
E
Customer.PreferredEmployeeID = Employee.ID
S
Employee.City = Supplier.City
I
Supplier.ID = Item.SupplierID
AND
Order.OrderDate > Item.DiscontinuedDate
922
O
I
Order.OrderDate > Item.DiscontinuedDate
(Order.Customer.PreferredEmployee.City == Item.Supplier.City)
AND
(Order.OrderDate > Item.DiscontinuedDate)
950
O
C
Order.CustomerID = Customer.ID
E
Customer.PreferredEmployeeID = Employee.ID
S
Employee.City = Supplier.City
I
(Supplier.ID = Item.SupplierID)
AND
(Order.OrderDate > Item.DiscontinuedDate)
Order.Customer.City == Item.Warehouse.City
952
O
C
Order.CustomerID = Customer.ID
W
Customer.City = Warehouse.City
I
Warehouse.ID = Item.WarehouseID
OR
((Order.Customer.PreferredEmployee.City == Item.Supplier.City) AND (Order.OrderDate > Item.DiscontinuedDate))
OR
(Order.Customer.City == Item.Warehouse.City)
954
O
C
Order.CustomerID = Customer.ID
W
Customer.City = Warehouse.City
E
Customer.PreferredEmployeeID = Employee.ID
S
Employee.City = Supplier.City
I
((Supplier.ID = Item.SupplierID) AND (Order.OrderDate > Item.DiscontinuedDate)) OR
(Warehouse.ID = Item.WarehouseID)

FIG. 13

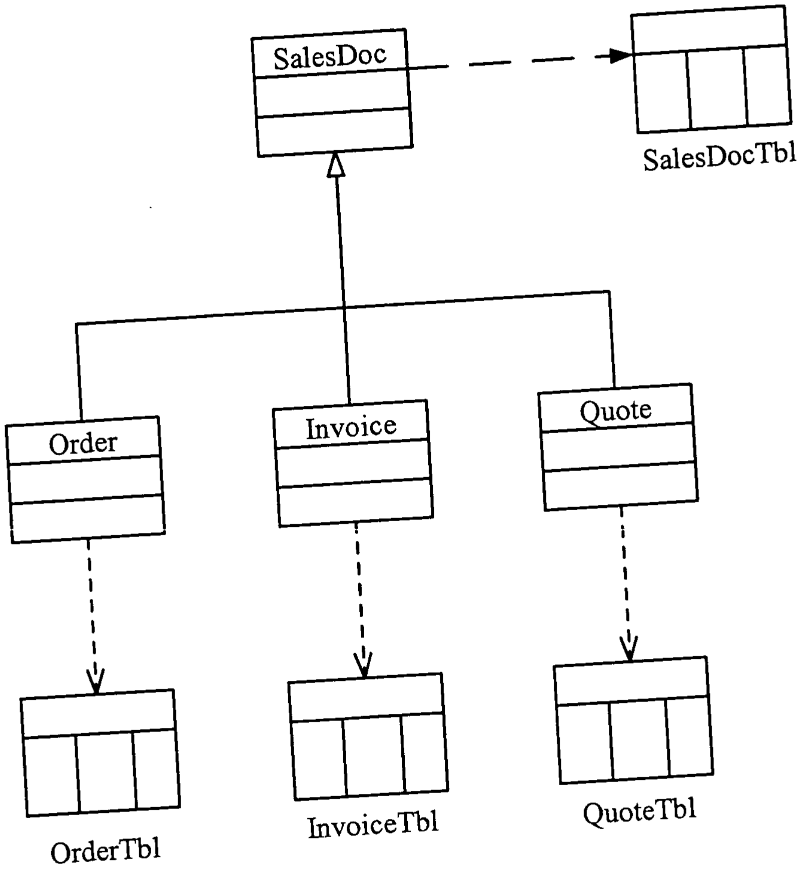

SalesDoc
SalesDocTbl
Order
Invoice
Quote
OrderTbl
InvoiceTbl
QuoteTbl

FIG. 14

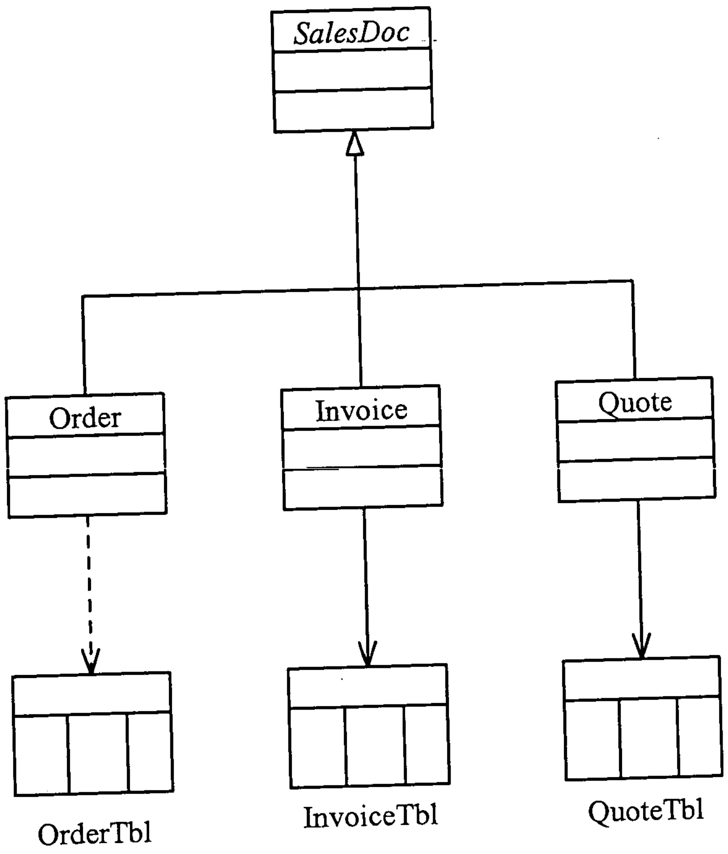

SalesDoc
Order
Invoice
Quote
OrderTbl
InvoiceTbl
QuoteTbl

FIG. 15

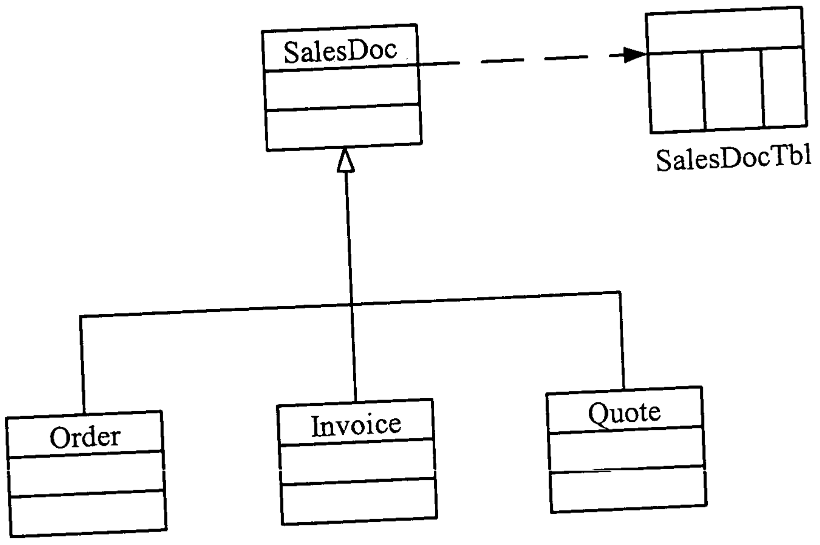


FIG. 16

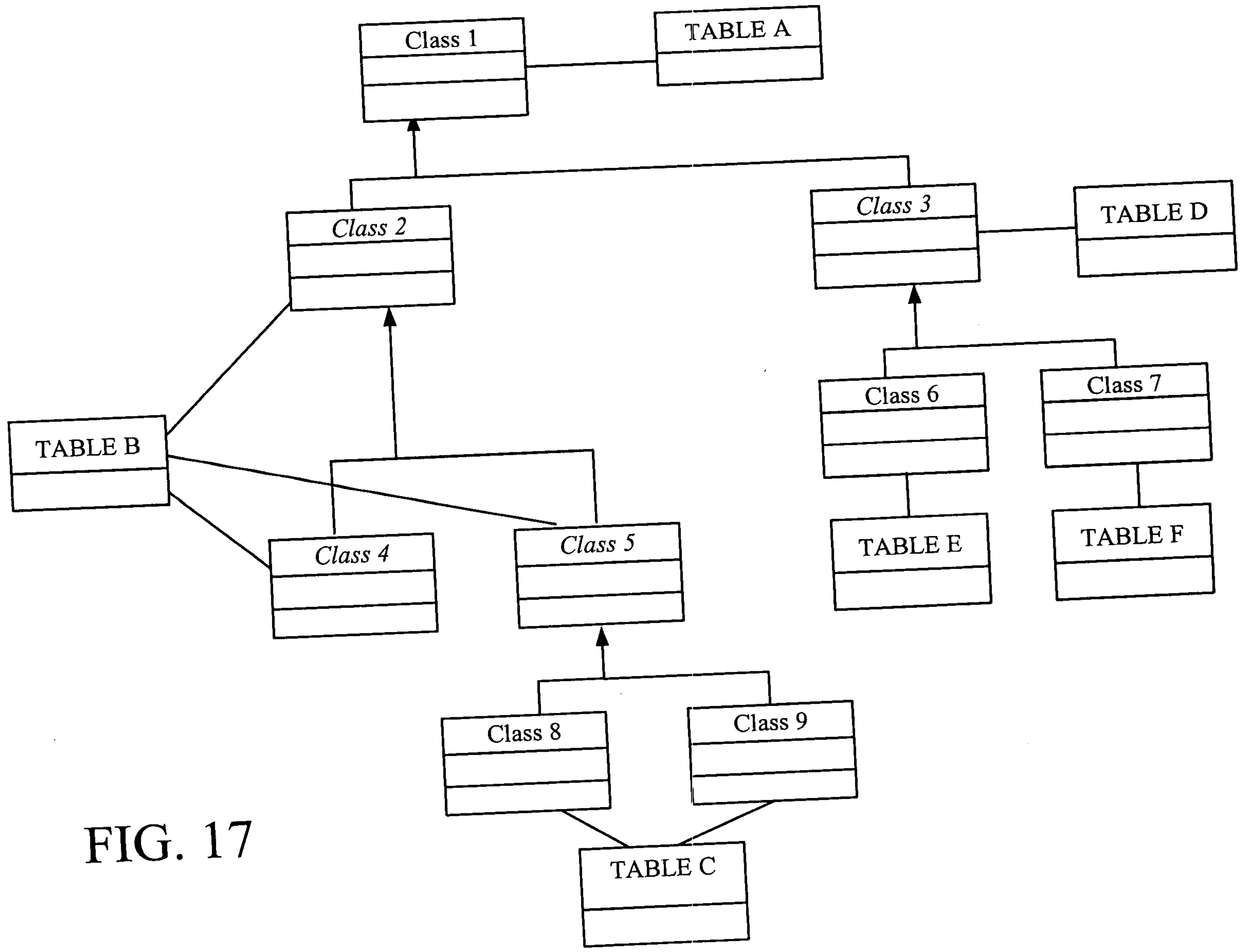
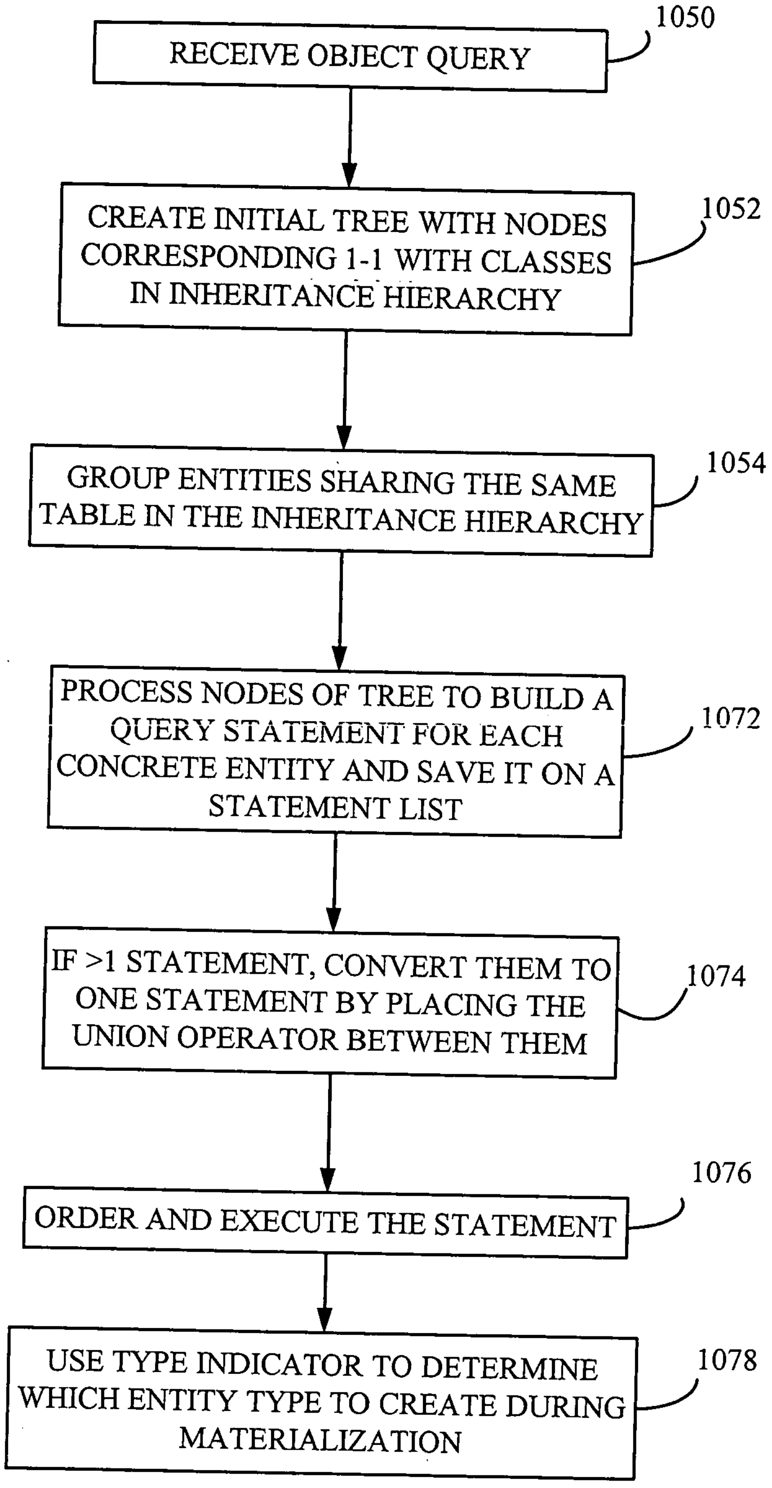

Class 1
TABLE A
Class 2
Class 3
TABLE D
TABLE B
Class 6
Class 7
Class 4
Class 5
TABLE E
TABLE F
Class 8
Class 9
TABLE C

FIG. 17



1050
RECEIVE OBJECT QUERY
1052
CREATE INITIAL TREE WITH NODES CORRESPONDING 1-1 WITH CLASSES IN INHERITANCE HIERARCHY
1054
GROUP ENTITIES SHARING THE SAME TABLE IN THE INHERITANCE HIERARCHY
1072
PROCESS NODES OF TREE TO BUILD A QUERY STATEMENT FOR EACH CONCRETE ENTITY AND SAVE IT ON A STATEMENT LIST
1074
IF >1 STATEMENT, CONVERT THEM TO ONE STATEMENT BY PLACING THE UNION OPERATOR BETWEEN THEM
1076
ORDER AND EXECUTE THE STATEMENT
1078
USE TYPE INDICATOR TO DETERMINE WHICH ENTITY TYPE TO CREATE DURING MATERIALIZATION

FIG. 18

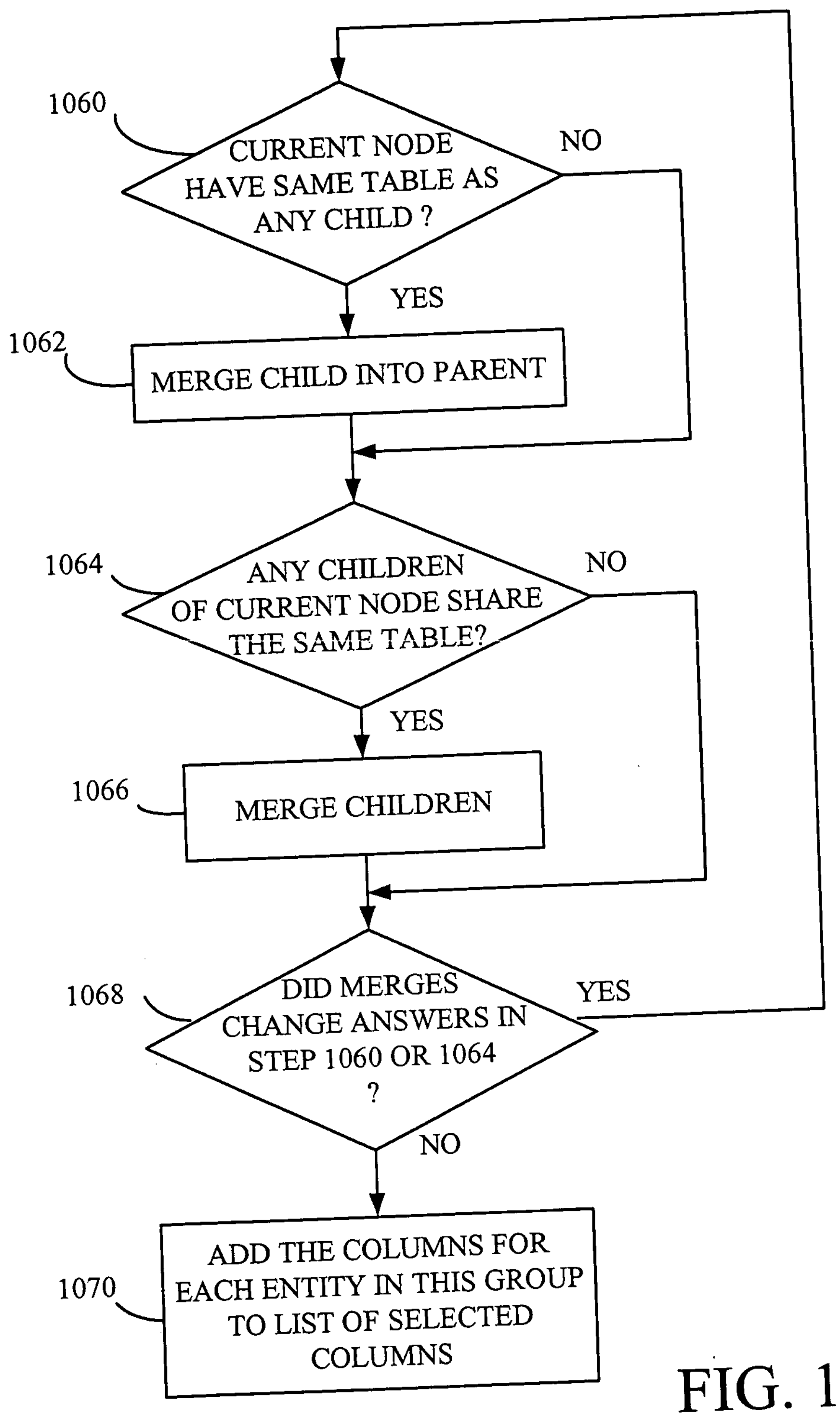

1060
CURRENT NODE HAVE SAME TABLE AS ANY CHILD ?
NO
YES
1062
MERGE CHILD INTO PARENT
1064
ANY CHILDREN OF CURRENT NODE SHARE THE SAME TABLE?
NO
YES
1066
MERGE CHILDREN
1068
DID MERGES CHANGE ANSWERS IN STEP 1060 OR 1064 ?
YES
NO
1070
ADD THE COLUMNS FOR EACH ENTITY IN THIS GROUP TO LIST OF SELECTED COLUMNS

FIG. 18-1

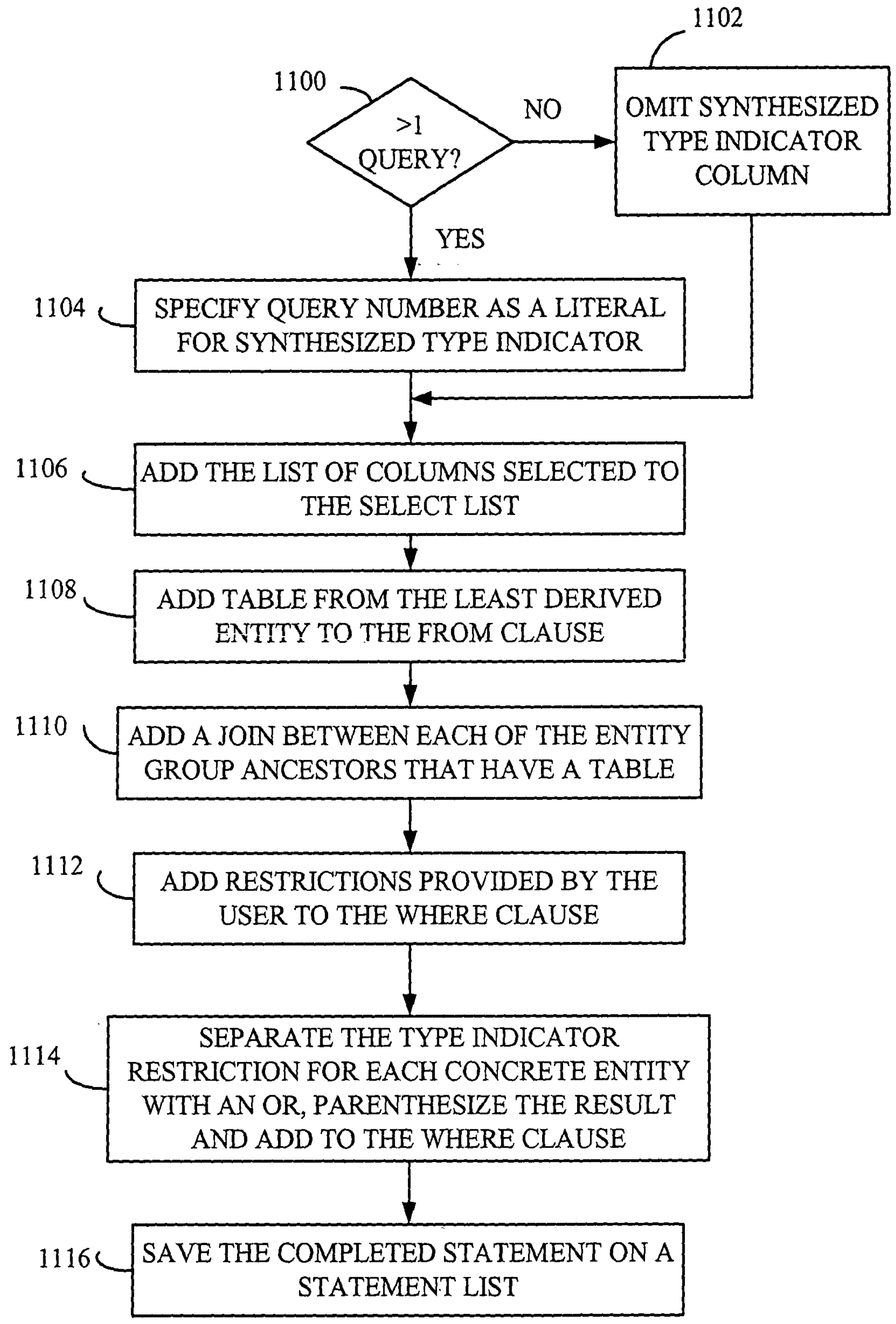

1102
1100
>1 QUERY?
NO
OMIT SYNTHESIZED TYPE INDICATOR COLUMN
YES
1104
SPECIFY QUERY NUMBER AS A LITERAL FOR SYNTHESIZED TYPE INDICATOR
1106
ADD THE LIST OF COLUMNS SELECTED TO THE SELECT LIST
1108
ADD TABLE FROM THE LEAST DERIVED ENTITY TO THE FROM CLAUSE
1110
ADD A JOIN BETWEEN EACH OF THE ENTITY GROUP ANCESTORS THAT HAVE A TABLE
1112
ADD RESTRICTIONS PROVIDED BY THE USER TO THE WHERE CLAUSE
1114
SEPARATE THE TYPE INDICATOR RESTRICTION FOR EACH CONCRETE ENTITY WITH AN OR, PARENTHESIZE THE RESULT AND ADD TO THE WHERE CLAUSE
1116
SAVE THE COMPLETED STATEMENT ON A STATEMENT LIST

FIG. 18-2

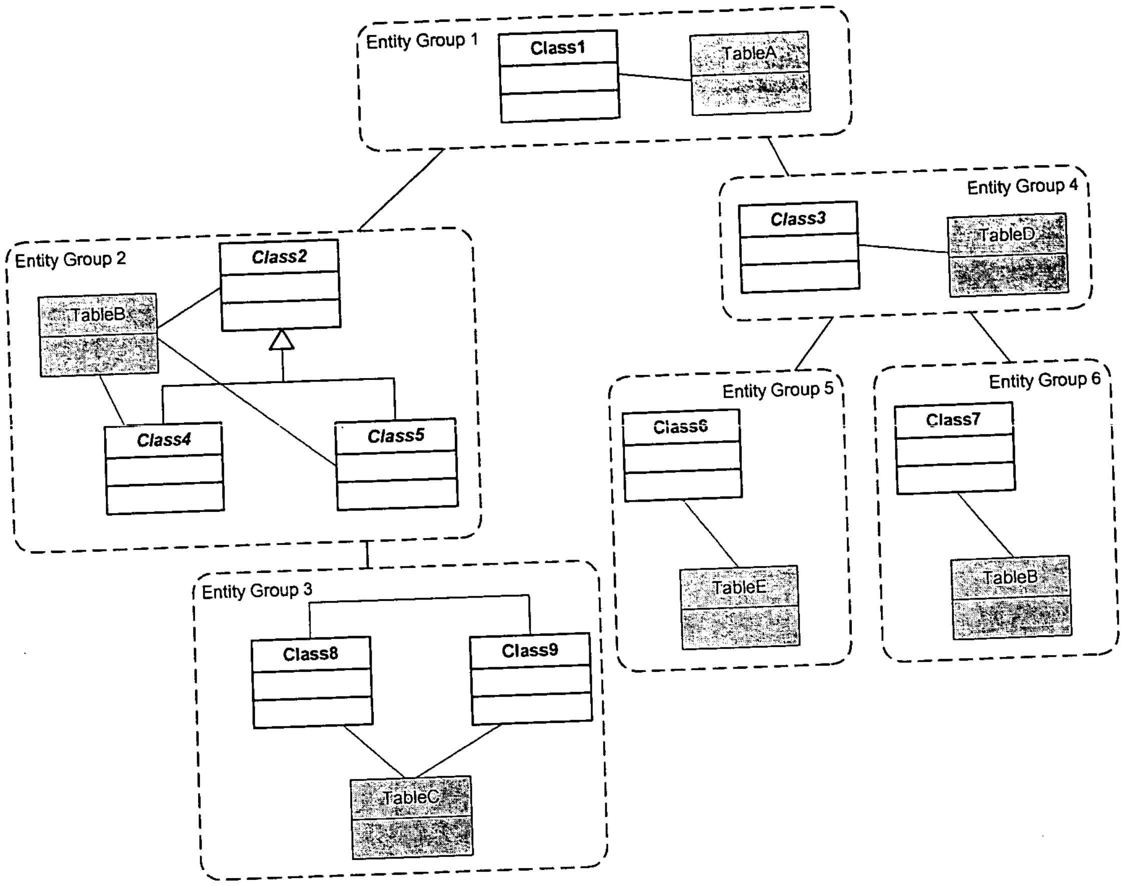

Entity Group 1
Class1
TableA
Entity Group 2
Class2
TableB
Class4
Class5
Entity Group 3
Class8
Class9
TableC
Entity Group 4
Class3
TableD
Entity Group 5
Class6
TableE
Entity Group 6
Class7
TableB

FIG. 19

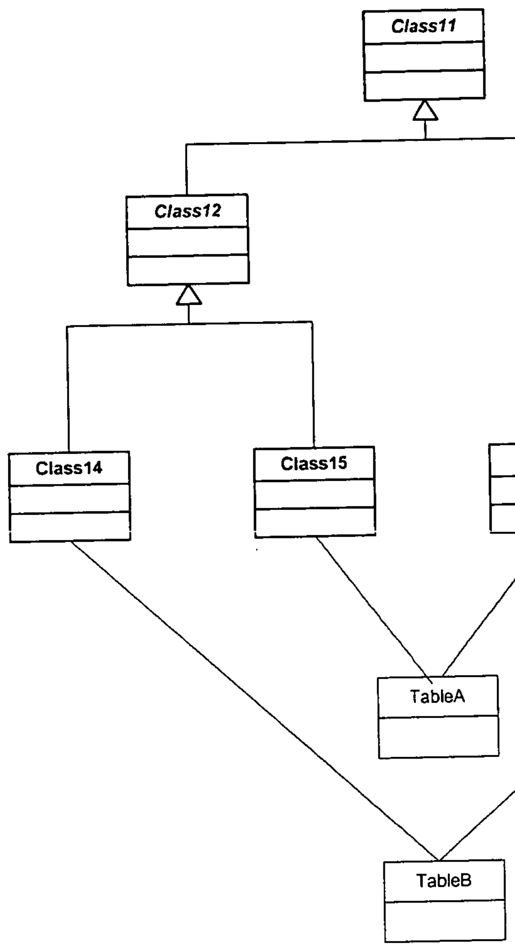

Class11
Class12
Class13
Class14
Class15
Class16
Class17
TableA
TableB

FIG. 20

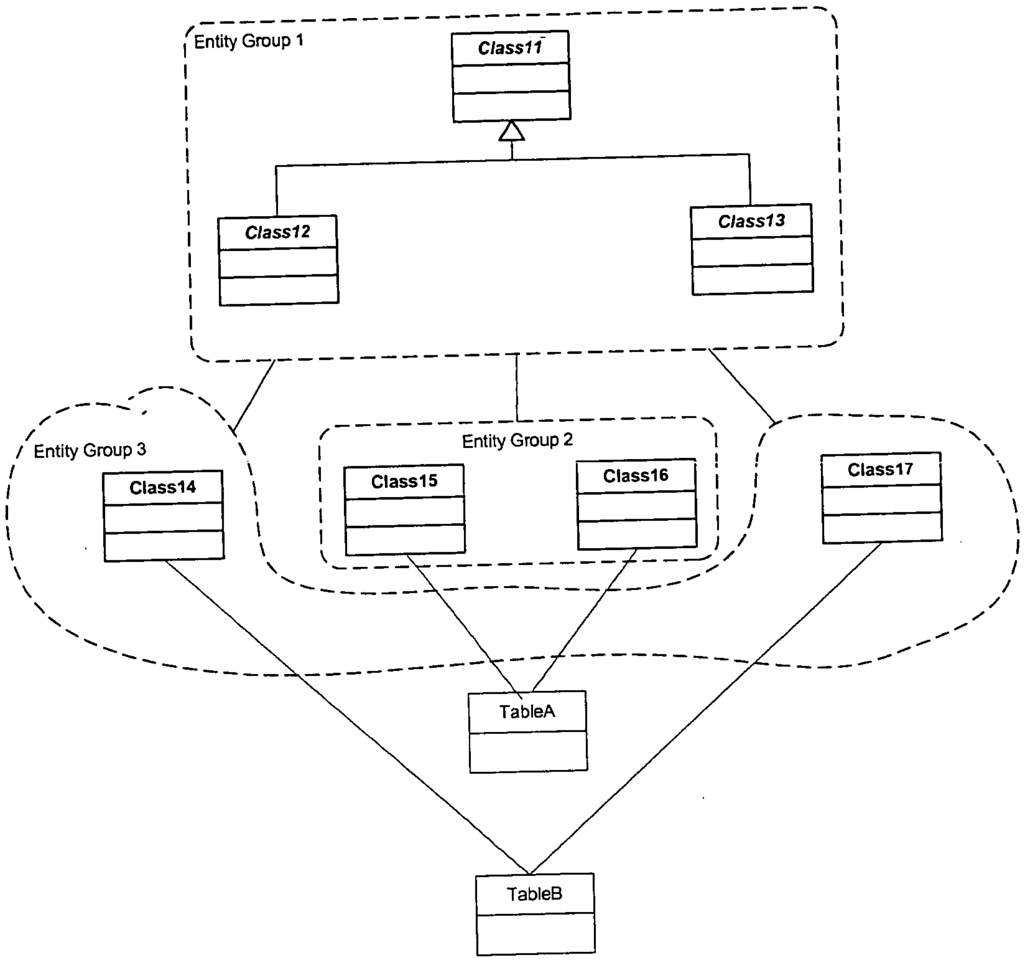
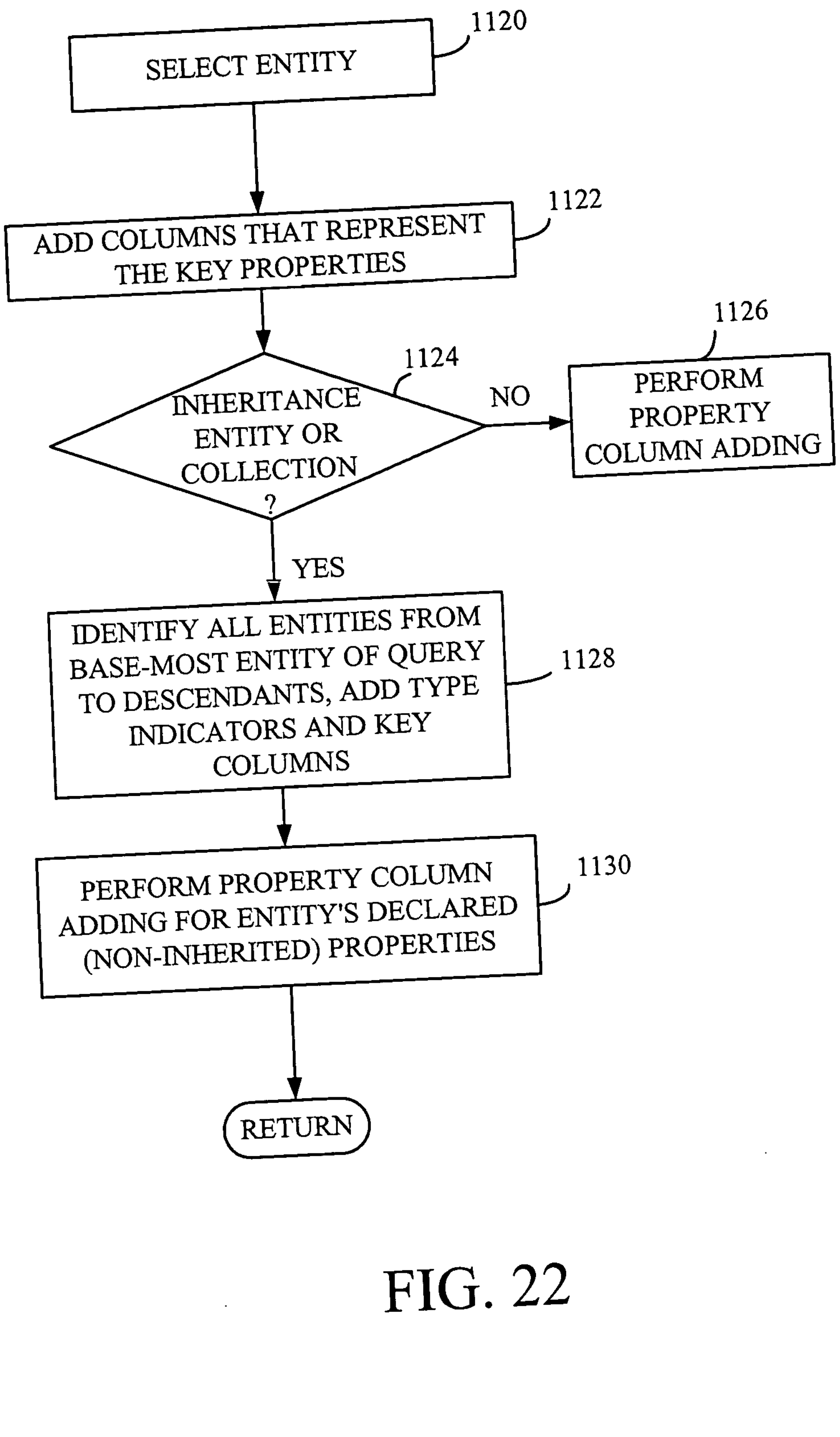
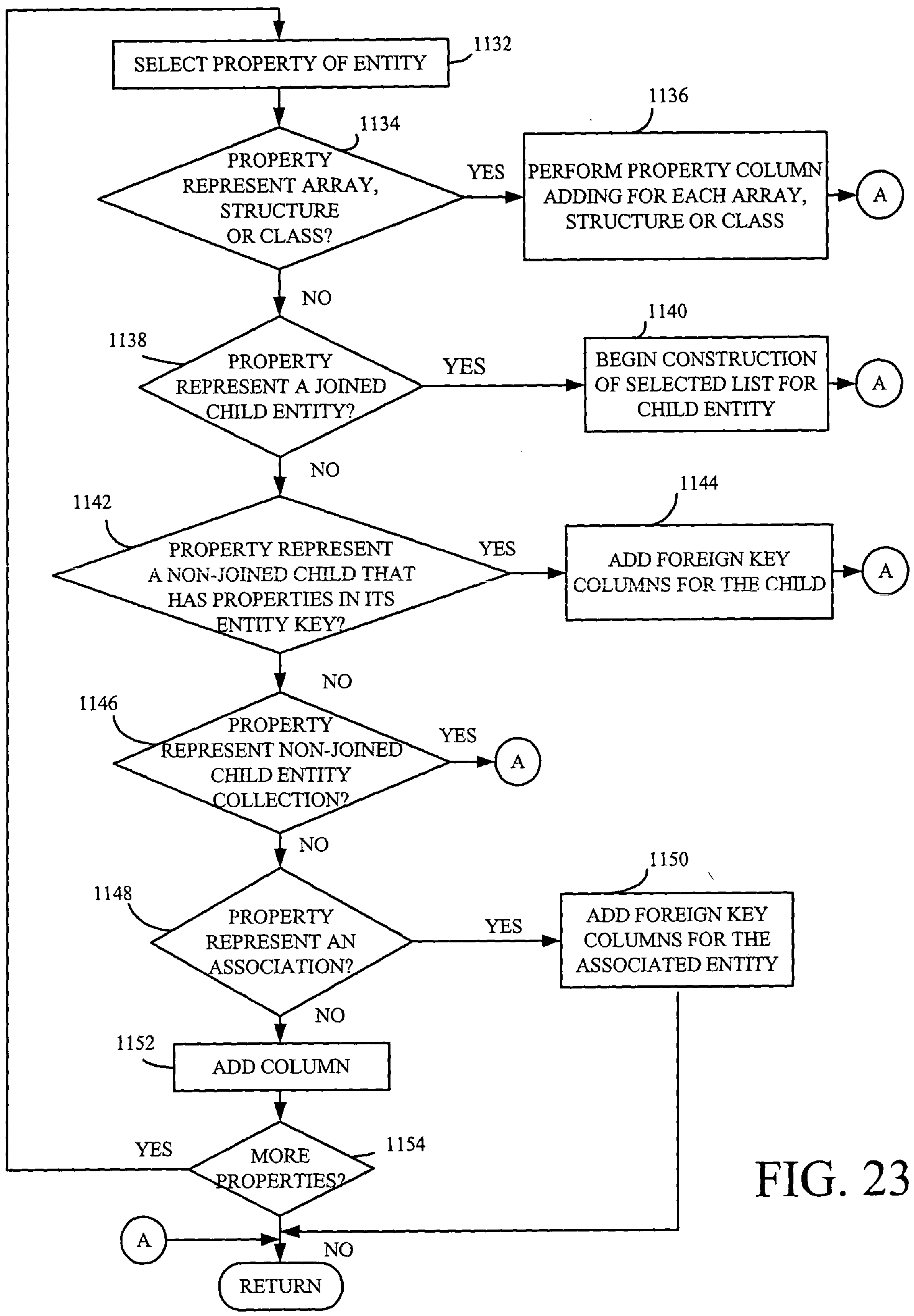

Entity Group 1
Class11
Class12
Class13
Entity Group 3
Entity Group 2
Class14
Class15
Class16
Class17
TableA
TableB

FIG. 21



1120
SELECT ENTITY
1122
ADD COLUMNS THAT REPRESENT THE KEY PROPERTIES
1124
INHERITANCE ENTITY OR COLLECTION ?
NO
1126
PERFORM PROPERTY COLUMN ADDING
YES
IDENTIFY ALL ENTITIES FROM BASE-MOST ENTITY OF QUERY TO DESCENDANTS, ADD TYPE INDICATORS AND KEY COLUMNS
1128
PERFORM PROPERTY COLUMN ADDING FOR ENTITY'S DECLARED (NON-INHERITED) PROPERTIES
1130
RETURN

FIG. 22



SELECT PROPERTY OF ENTITY
1132
1134
PROPERTY REPRESENT ARRAY, STRUCTURE OR CLASS?
YES
1136
PERFORM PROPERTY COLUMN ADDING FOR EACH ARRAY, STRUCTURE OR CLASS
A
NO
1138
PROPERTY REPRESENT A JOINED CHILD ENTITY?
YES
1140
BEGIN CONSTRUCTION OF SELECTED LIST FOR CHILD ENTITY
A
NO
1142
PROPERTY REPRESENT A NON-JOINED CHILD THAT HAS PROPERTIES IN ITS ENTITY KEY?
YES
1144
ADD FOREIGN KEY COLUMNS FOR THE CHILD
A
NO
1146
PROPERTY REPRESENT NON-JOINED CHILD ENTITY COLLECTION?
YES
A
NO
1148
PROPERTY REPRESENT AN ASSOCIATION?
YES
1150
ADD FOREIGN KEY COLUMNS FOR THE ASSOCIATED ENTITY
NO
1152
ADD COLUMN
YES
MORE PROPERTIES?
1154
A
NO
RETURN

FIG. 23

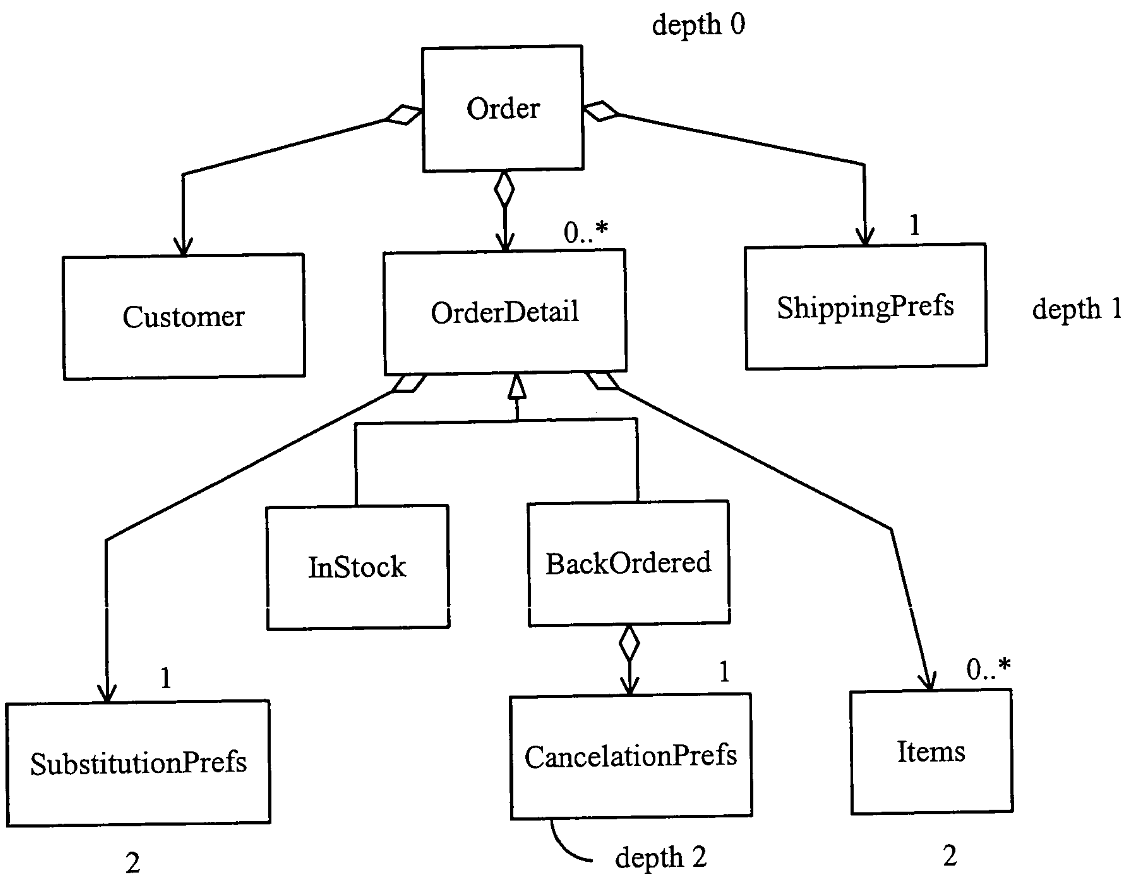
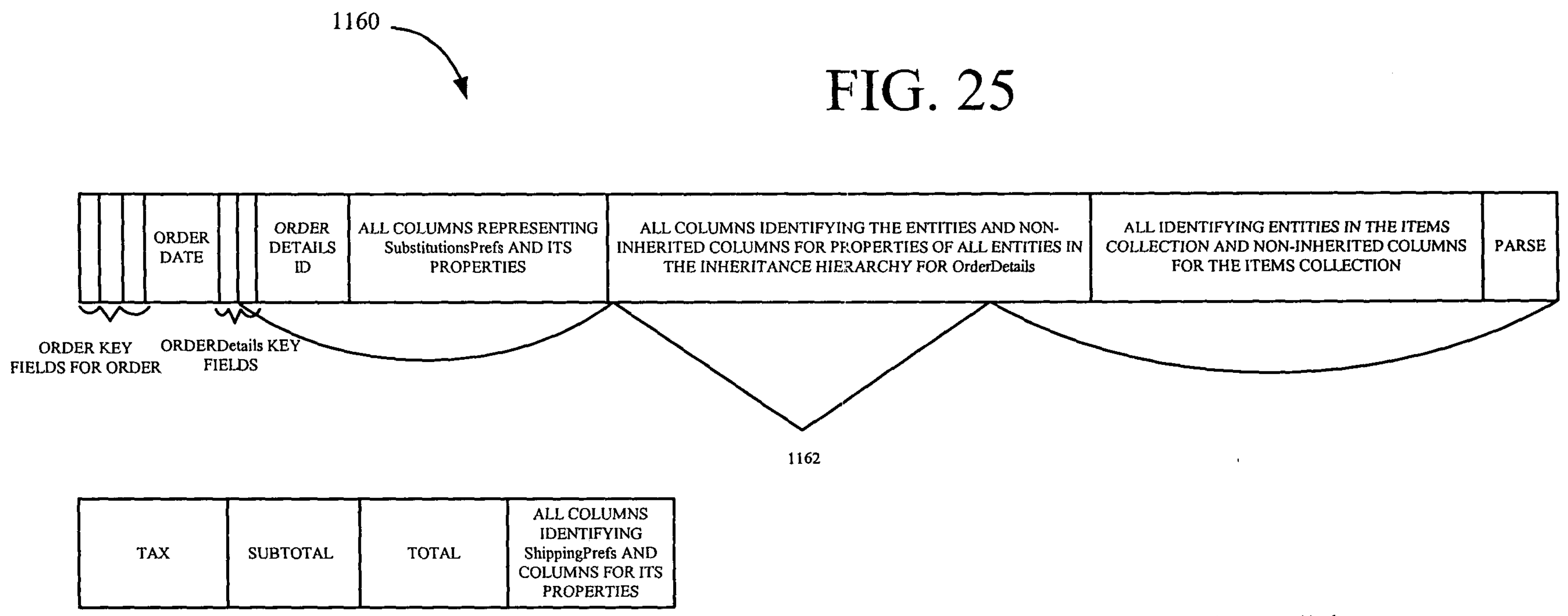

depth 0
Order
0..*
1
Customer
OrderDetail
ShippingPrefs
depth 1
InStock
BackOrdered
1
1
0..*
SubstitutionPrefs
CancelationPrefs
Items
2
depth 2
2

FIG. 24

FIG. 25


1160
ORDER DATE
ORDER DETAILS ID
ALL COLUMNS REPRESENTING SubstitutionsPrefs AND ITS PROPERTIES
ALL COLUMNS IDENTIFYING THE ENTITIES AND NON-INHERITED COLUMNS FOR PROPERTIES OF ALL ENTITIES IN THE INHERITANCE HIERARCHY FOR OrderDetails
ALL IDENTIFYING ENTITIES IN THE ITEMS COLLECTION AND NON-INHERITED COLUMNS FOR THE ITEMS COLLECTION
PARSE
ORDER KEY FIELDS FOR ORDER
ORDERDetails KEY FIELDS
1162
TAX
SUBTOTAL
TOTAL
ALL COLUMNS IDENTIFYING ShippingPrefs AND COLUMNS FOR ITS PROPERTIES

```
CLASS ORDER {
        ID
        DATE
        OrderDDetails
        Collection
        Tax
        Subtotal
        Total
        ShippingPrefs
}
```

FIG. 25A

```
CLASS OrderDetail {
        ID
        SubstitutionP
        refs
        Items
        Collection
        Misc
}
```

FIG. 25B

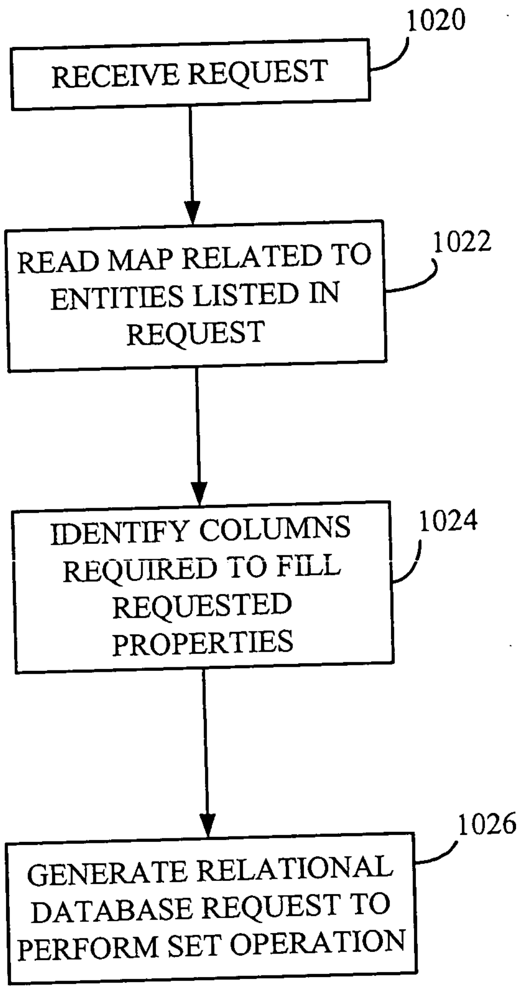
212

```
1002    class Order {          // these fields are mapped to a database table public string ID;
            ...
            public DetailCollection Details;   // contains Detail objectspublic decimal Total;}
        class Detail {        // these fields are mapped to a database table
1004        public long SequenceNumber;
            public Item Item;
            public decimal PricePerUnit;
            public decimal Quantity;
            public decimal Price;                     // PricePerUnit * Quantity}

1006    // the user describes the set operation they want performed in terms of objects
        Criteria.EntitySetUpdateCriteria(Criteria.EntityAlias(parentKey, typeof(Order))),
1010    // update the order// set Order.Total to the sum of each of the line item's price
            Criteria.PropertyAssignments(Criteria.Assignment((Property)"Order.Total",
        Criteria.Sum((Property)"Order.Details[].Price")      )
1008    // indicates which orders to update; only those with detail price > 300
            Criteria.Where((Property)"Order.OrderDetails[].Price" > 300),)
```

FIG. 26

FIG. 27



1020
RECEIVE REQUEST
1022
READ MAP RELATED TO ENTITIES LISTED IN REQUEST
1024
IDENTIFY COLUMNS REQUIRED TO FILL REQUESTED PROPERTIES
1026
GENERATE RELATIONAL DATABASE REQUEST TO PERFORM SET OPERATION

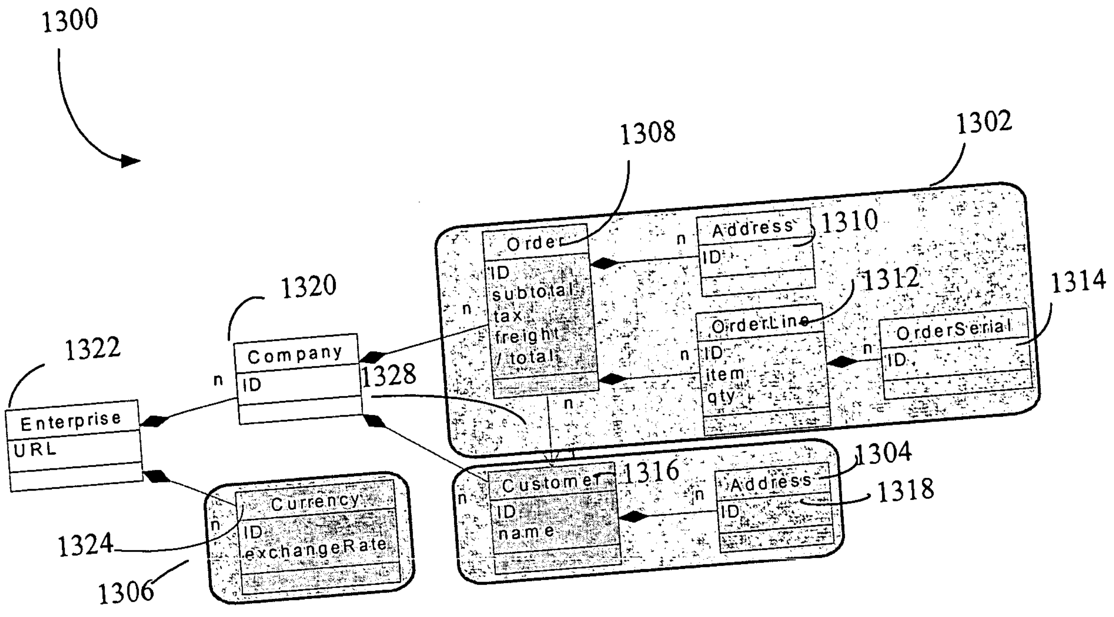

1300
1302
1308
Order
ID
subtotal
tax
freight
/ total
Address
ID
1310
1312
OrderLine
ID
item
qty
OrderSerial
ID
1314
1320
Company
ID
1322
Enterprise
URL
1328
Customer
ID
name
1316
Address
ID
1304
1318
Currency
ID
exchangeRate
1324
1306
n

FIG. 28

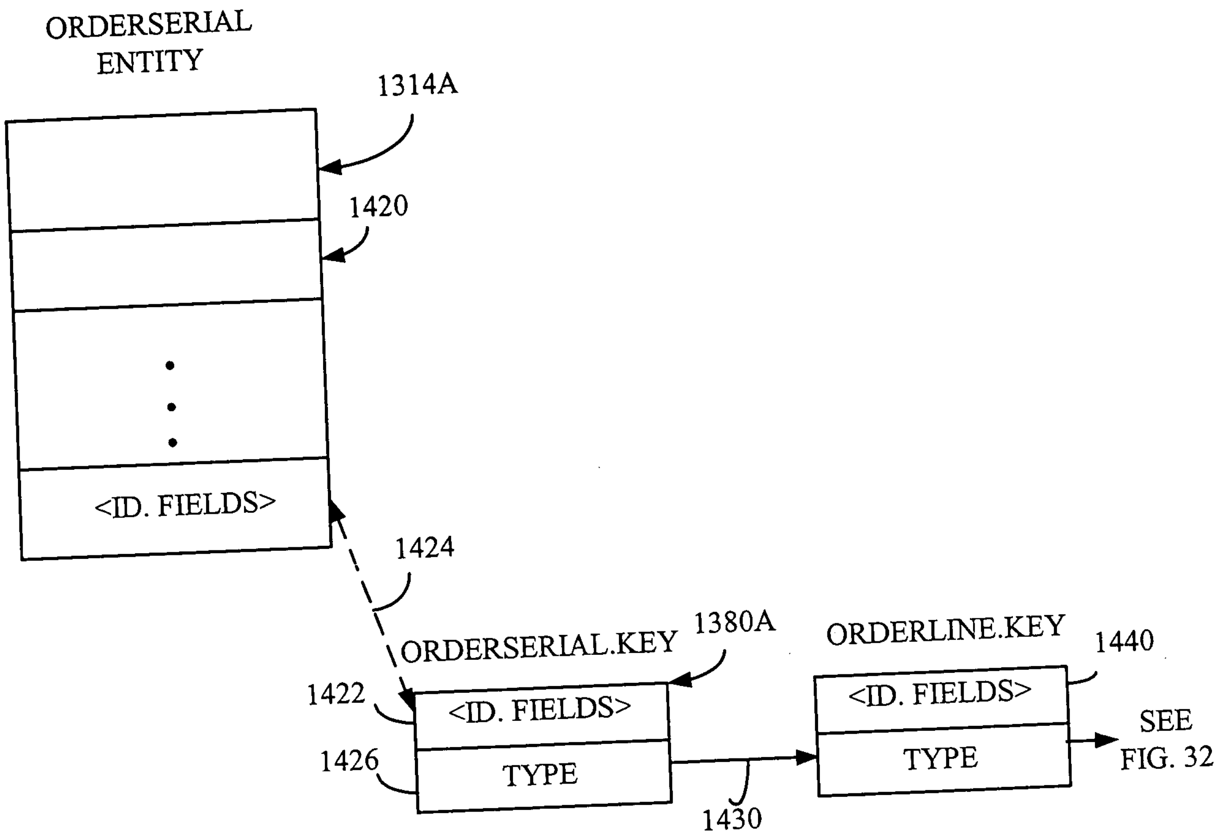

ORDERSERIAL ENTITY
1314A
1420
<ID. FIELDS>
1424
ORDERSERIAL.KEY
1380A
ORDERLINE.KEY
1440
1422
<ID. FIELDS>
<ID. FIELDS>
1426
TYPE
TYPE
SEE FIG. 32
1430

FIG. 29

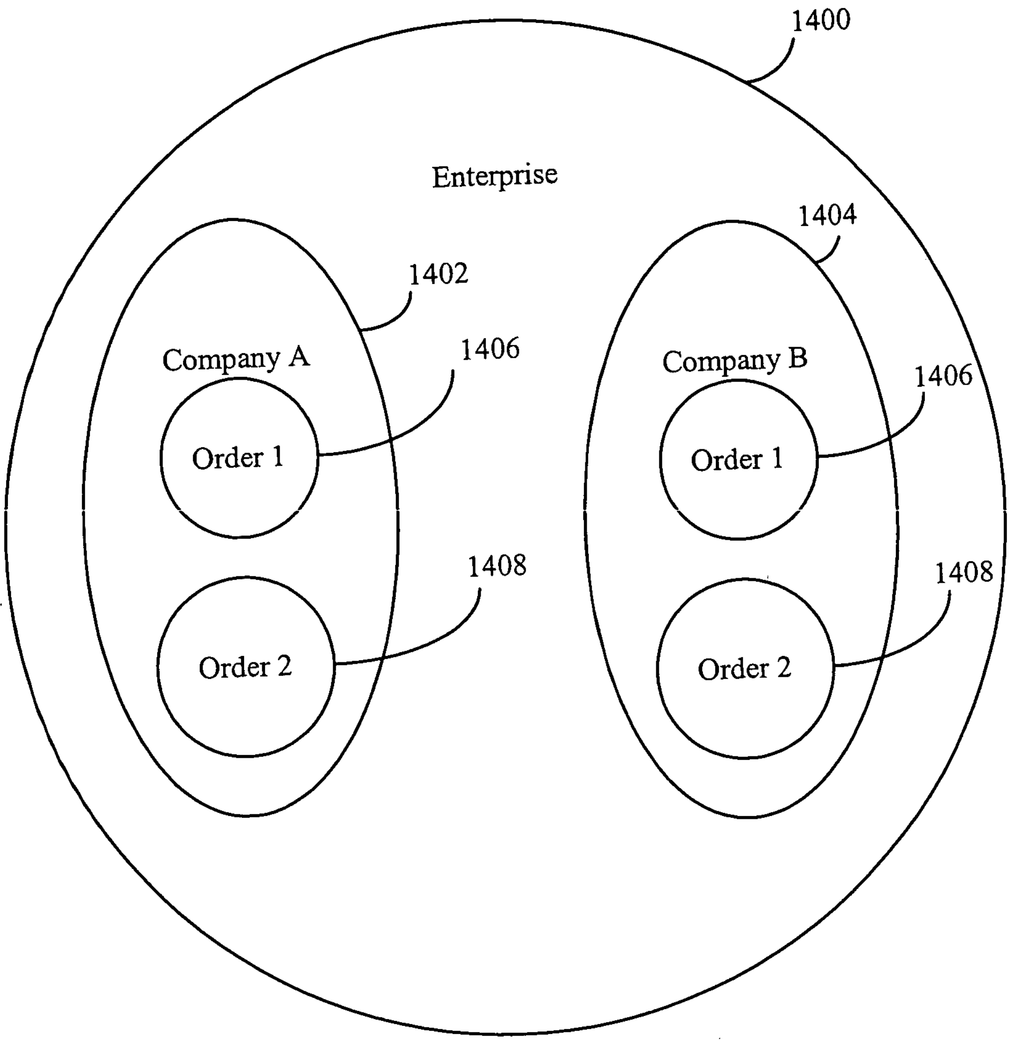

1400
Enterprise
1404
1402
Company A
1406
Company B
1406
Order 1
Order 1
1408
1408
Order 2
Order 2

FIG. 30

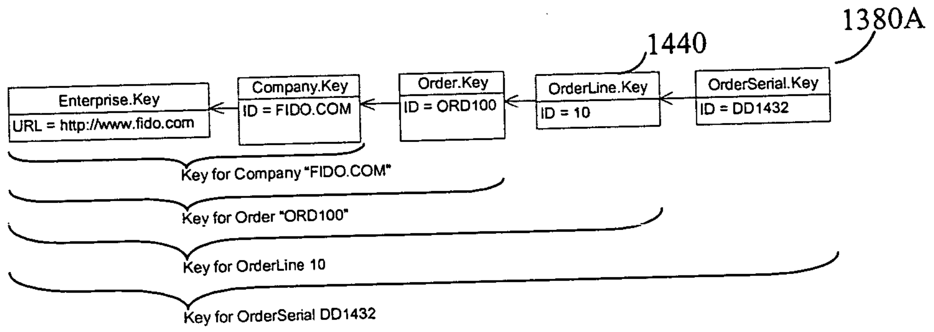

1440
1380A
Enterprise.Key
URL = http://www.fido.com
Company.Key
ID = FIDO.COM
Order.Key
ID = ORD100
OrderLine.Key
ID = 10
OrderSerial.Key
ID = DD1432
Key for Company "FIDO.COM"
Key for Order "ORD100"
Key for OrderLine 10
Key for OrderSerial DD1432

FIG. 31

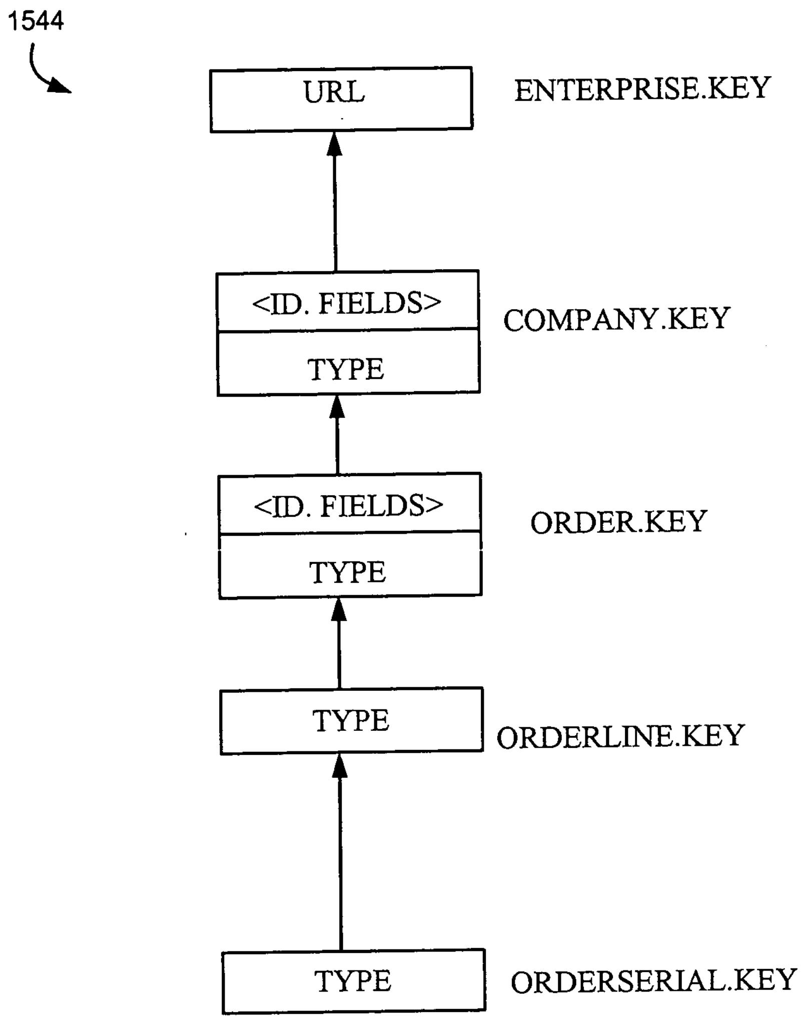

1544
URL
ENTERPRISE.KEY
<ID. FIELDS>
TYPE
COMPANY.KEY
<ID. FIELDS>
TYPE
ORDER.KEY
TYPE
ORDERLINE.KEY
TYPE
ORDERSERIAL.KEY

FIG. 32

1350

COMPANY_ID 1352	ORDER_ID 1354	ORDERLINE_ID 1356	SERIAL NO. 1358	OTHER COLUMNS	
⋮	⋮	⋮	⋮	⋮	⋮

FIG. 33